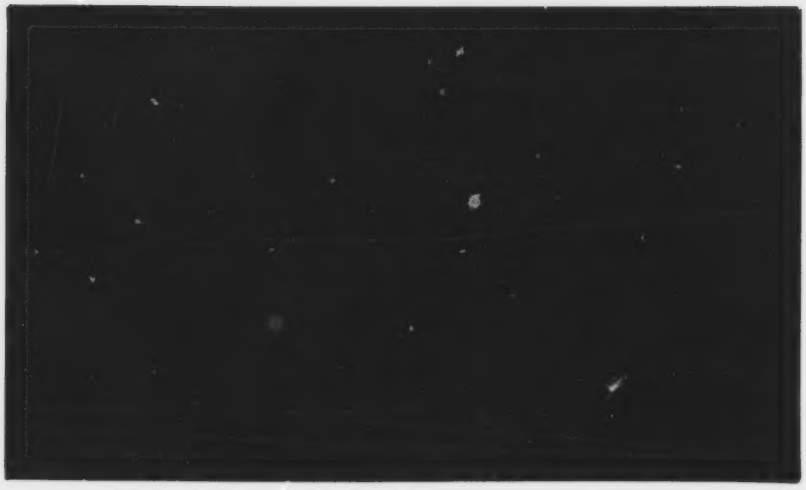
CENTRE
DE RECHERCHE
SUR LES TRANSPORTS
(CRT)
CENTRE
FOR RESEARCH
ON TRANSPORTATION

UNIVERSITÉ DE MONTRÉAL
ÉCOLE DES HAUTES ÉTUDES COMMERCIALES
ÉCOLE POLYTECHNIQUE



MÉMOIRE D'ÉTUDIANT

C.P. 6128, SUCCURSALE A
MONTRÉAL (QUÉBEC)
CANADA
H3C 3J7

TÉLÉPHONE : (514) 343-7575
TÉLEX : 05-24146 BIBPOLYTEC
TÉLÉCOPIEUR (FAX) : (514) 343-7121

CRT-646

CONTRIBUTIONS AU DÉVELOPPEMENT D'UN SIMULATEUR INTERACTIF-GRAPHIQUE D'ENTREPRISES DE TRANSPORT INTERURBAIN DE PASSAGERS

par

Johanne Dion*

* Département d'Informatique et de Recherche opérationnelle, Université de Montréal.

Rapport de stage effectué au Centre de recherche sur les transports et présenté à la Faculté des Études supérieures en vue de l'obtention du grade de Maîtrise es Sciences (Informatique).

Université de Montréal

Centre de recherche sur les transports - Publication #646

Mai 1989

89-05951/

Dépôt légal - 3e trimestre 1989

SOMMAIRE

Le transport interurbain de passagers est une activité des plus importante. La planification et la gestion d'une compagnie opérant dans ce domaine sont des tâches très complexes. Cette complexité intervient d'abord au niveau de la quantité et de la diversité des décisions à prendre, ces dernières étant de plus très interreliées, ainsi qu'au niveau de la nécessité d'une compréhension profonde et détaillée de l'environnement dans lequel évolue la compagnie.

Le simulateur d'entreprise JET-SET est un outil d'analyse, d'expérimentation et de formation dans le domaine du transport interurbain de passagers. Sous la forme d'un jeu informatisé, il permet aux participants d'expérimenter les différents aspects de la planification et de la gestion d'une compagnie, ceci par l'intermédiaire d'une série de décisions à prendre, ainsi que par l'étude des résultats fournis par la simulation.

Ce rapport de stage porte sur l'intégration de la dimension opérationnelle au simulateur JET-SET. Plus précisément, cela implique l'ajout des décisions de type "routing-scheduling" au processus de prise de décision auquel sont confrontés les participants. Pour ce faire, nous avons développé un "système interactif-graphique de prise de décision" qui répartit le processus sur cinq départements décisionnels reproduisant ainsi une structure organisationnelle comparable à celles qu'on peut rencontrer dans de véritables compagnies. Le développement de ce système a exigé la création d'interfaces graphiques permettant la consultation et la manipulation des différentes données opérationnelles, soient les horaires de service, les horaires publiés et les routes-horaires. Ces données sont conservées dans une banque de données qui a été spécialement conçue pour un traitement interactif efficace.

Ce rapport traite aussi de la génération des connexions menant aux services indirects. Nous analysons la problématique de cette génération et nous proposons une méthode de résolution qui repose sur une subdivision des arrivées et des départs de services en une ville en plages horaires. Nous présentons ensuite un algorithme, basé sur cette méthode, qui permet une mise à jour instantanée des services indirects présentés dans les horaires publiés.

Mots-clés : simulation; transport interurbain de passagers; routes; horaires.

TABLE DES MATIÈRES

Page

SOMMAIRE iii

LISTE DES TABLEAUX viii

LISTE DES FIGURES ix

INTRODUCTION 1

CHAPITRE 1 : SITUATION DU PROBLÈME 4

1.1 Description de la problématique 4
1.1.1 Environnement 5
1.1.2 Processus de prise de décision 6
1.1.3 Approche de résolution 7

1.2 JET-SET : Un simulateur d'entreprises de transport interurbain de passagers 9

1.3 JET-SET/3 et la gestion des opérations 13

1.4 Objectifs du stage 14

1.5 Définition de termes 15

CHAPITRE 2 : DESIGN DU SIMULATEUR 18

2.1 Structure départementale 18

2.2 Déroulement d'une séance de simulation 22

2.3 Description des interfaces 25
2.3.1 Département Planification 25
2.3.1.1 Utilisation générale 28
2.3.1.1.1 Activer une zone 28
2.3.1.1.2 Changer la fenêtre courante 29
2.3.1.1.3 Choisir une commande 29
2.3.1.1.4 Remplir une fenêtre de données 29
2.3.1.1.5 Défiler une liste 30
2.3.1.1.6 Aide 30
2.3.1.1.7 Quitter 30
2.3.1.2 Manipulation des données 30
2.3.1.2.1 Zone de consultation 30
2.3.1.2.2 Zone de travail 31

TABLE DES MATIÈRES (suite)

Page

2.3.2 Département Gestion des opérations 32
2.3.2.1 Représentation graphique 32
2.3.2.2 Présentation des interfaces 35
2.3.2.1.1 Interface 1 35
2.3.2.1.2 Interface 2 42

CHAPITRE 3 : BANQUE DE DONNÉES 44

3.1 Design de la banque de données 44

3.2 Structure de la banque de données 45
3.2.1 Structure générale 45
3.2.2 Données 48
3.2.3 Composante de la structure 49
3.2.3.1 Horaire de service 49
3.2.3.2 Route-horaire 51
3.2.3.3 Liste d'activités 52
3.2.3.4 Horaire publié 55
3.2.3.5 Gestion interne 57
3.2.4 Intégration 59

3.3 Modifications envisagées 60

3.4 Expérimentation 62

CHAPITRE 4 : GÉNÉRATION DES CONNEXIONS 63

4.1 Description de la problématique 63
4.1.1 Choix des itinéraires 64
4.1.2 Choix du critère de connexion 65

4.2 Notion de plage horaire 68

4.3 Génération des connexions 71
4.3.1 Perspective dynamique 71
4.3.2 Algorithme 74
4.3.2.1 Principe général 74
4.3.2.2 Traitement des plages 77
4.3.2.3 Présentation algorithmique 80

CHAPITRE 5 : RÉALISATION DU LOGICIEL 88

5.1 Gestion de la banque de données 88
5.1.1 Module d'initialisation 89
5.1.2 Module d'insertion 90
5.1.3 Module d'enlèvement 92
5.1.4 Module de connexion 93
5.1.5 Module charger/sauver 94

TABLE DES MATIÈRES (suite)

Page

5.2 Gestion des interfaces 97
5.2.1 Outils logiciels 97
5.2.2 Département Planification 98
5.2.2.1 Module d'interception 99
5.2.2.2 Module de positionnement 99
5.2.2.3 Module de défilement 99
5.2.2.4 Module de changement de liste 100
5.2.2.5 Module d'insertion 100
5.2.2.6 Module d'enlèvement 100
5.2.2.7 Module de modification 101
5.2.2.8 Module de mise à jour 101
5.2.3 Département Gestion des opérations 101
5.2.3.1 Module du graphisme 102
5.2.3.2 Module d'interception 103
5.2.3.3 Module d'inspection 103
5.2.3.4 Module de défilement 104
5.2.3.5 Module de déplacement 104
5.2.3.6 Module d'insertion 105
5.2.3.7 Module d'enlèvement 106
5.2.3.8 Module de croisement 106
5.2.3.9 Module de gestion de l'horaire 107
5.2.3.10 Module de graphique ESP-TPS 107

CONCLUSION 109

RÉFÉRENCES 111

APPENDICE A : DÉCLARATION DE LA STRUCTURE
DE LA BANQUE DE DONNÉES 113

APPENDICE B : EXTRAITS DU CODE 116

REMERCIEMENTS 124

LISTE DES TABLEAUX

Page

Tableau I : Modules de gestion de la banque de données 89
Tableau II : Structure des fichiers 95
Tableau III : Modules du département Planification 99
Tableau IV : Modules du département Gestion des opérations 102
Tableau V : Fonctions de l'interface de consultation 107
Tableau VI : Variables 117
Tableau VII : Constantes 117

LISTE DES FIGURES

Page

Figure 2.1 : Processus d'élaboration de l'horaire opérationnel effectif 23

Figure 2.2 : Interface du département Planification 27

Figure 2.3 : Diagramme véhicule-temps de routes-horaires 33

Figure 2.4 : Interface 1 du département Gestion des opérations . . . 36

Figure 2.5 : Croisement de routes-horaires 39

Figure 2.6 : Interface de consultation 40

Figure 2.7 : Interface 2 du département Gestion des opérations . . . 42

Figure 3.1 : Structure des horaires de service 50

Figure 3.2 : Structure des routes-horaires 52

Figure 3.3 : Structure des listes d'activités 54

Figure 3.4 : Structure (partielle) des horaires publiés 56

Figure 3.5 : Liens existants entre services et services indirects . . 59

Figure 3.6 : Structure de multiliste de la banque de données 61

Figure 4.1 : Schéma d'arrivées et de départs (A/D) 66

Figure 4.2 : Plages horaires 69

Figure 4.3 : Schéma A/D balancé 70

Figure 4.4 : Schémas A/D contenant un nouveau service 71

Figure 4.5 : Schémas A/D avec enlèvement de service 73

Figure 4.6 : Schémas A/D avec insertion de service 74

Figure 4.7 : Liste d'activités à la ville B 76

Figure 4.8 : Schéma A/D avec plage généralisée 77

Figure 4.9 : Schémas A/D séparés 79

Figure 4.10 : Étapes du processus de génération 81

Figure 4.11 : Combinaison de services indirects 85

INTRODUCTION

Le problème de la planification et de la gestion du transport interurbain de passagers est très complexe. Les planificateurs/gestionnaires doivent prendre de multiples décisions, très diverses et interreliées. Cette prise de décision intervient dans un contexte de marché concurrentiel et elle doit tenir compte de l'environnement social, économique, technologique et réglementaire.

Une question fondamentale se pose, à savoir comment former des individus pour une profession nécessitant une telle expertise.

Les simulateurs d'entreprises se veulent une méthode efficace d'apprentissage de situations comparables à celles rencontrées en pratique tout en pouvant servir de laboratoire expérimental pour l'étude de différentes stratégies d'opération dans les compagnies concernées. JET-SET (Jeu Educatif en Transport - Simulation for Education in Transportation), développé au Centre de recherche sur les transports (C.R.T.) de l'Université de Montréal, constitue un exemple de ce type de simulateur dans le domaine du transport. Il est conçu sous la forme d'un jeu informatisé dans lequel un certain nombre de participants possèdent et gèrent chacun une compagnie de transport interurbain de passagers évoluant dans un environnement de marché compétitif. En tant qu'outil de formation, il permet d'expérimenter toute la complexité et la diversité des situations et décisions rencontrées dans la réalité.

Le stage qui fait l'objet de ce rapport a porté sur le développement d'une troisième version du simulateur JET-SET. Il s'agissait d'intégrer la dimension "routing-scheduling" au processus d'analyse et de décision auquel sont confrontés les participants. Cette dimension comprend d'une part les *horaires de service* de la compagnie, c'est-à-dire la liste ordonnée des

services pour chaque paire de villes du réseau physique désservi, et d'autre part la description détaillée des routes effectuées par chaque véhicule; nous désignons celles-ci sous le terme *route-horaire*. Un *service* représente ici un transport de passagers d'une ville à une autre assuré par un véhicule de la compagnie.

Plus spécifiquement, l'objectif premier du stage a été de concevoir un système interactif-graphique permettant l'élaboration des horaires de service et des routes-horaires à l'écran d'un micro-ordinateur. La réalisation de cet objectif a nécessité une restructuration complète du simulateur. En particulier, une structure de données plus complexe a dû être élaborée pour conserver les horaires de service et les routes-horaires, ainsi que pour permettre un traitement dynamique et interactif de ces informations. De plus, la conception des interfaces pour l'entrée interactive des décisions a nécessité une révision intégrale du module d'entrée du simulateur.

Au terme du stage, nous avons obtenu le prototype opérationnel d'un tout nouveau simulateur incluant un **système interactif-graphique de prise de décision.** Dans ce système, les décisions sont réparties entre cinq *départements* décisionnels. La notion organisationnelle de département permet de différencier, à l'intérieur du simulateur, les différents domaines d'application des décisions, soient la planification, la gestion des opérations, le marketing, les achats et locations et finalement les finances.

Parmi ces cinq départements, les deux premiers sont pleinement développés. Celui de la Planification permet la construction des horaires de service ainsi que la génération automatique des *horaires publiés*, c'est-à-dire les horaires que consultent les voyageurs et qui comportent non seulement les services proprement dits, mais aussi les services indirects formés par des *connexions* entre les services. Le département Gestion des opérations, quant à lui, permet l'élaboration des

routes-horaires des véhicules. Ces prises de décision s'effectuent à l'aide d'interfaces numériques et graphiques qui offrent plusieurs outils d'aide à la décision.

Le système interactif-graphique de prise de décision interagit avec une banque de données contenant les horaires de service, les routes-horaires, les horaires publiés et quelques autres données utilitaires. Sa structure permet la consultation et la mise à jour rapide des données. Elle est aussi spécialement conçue pour certains traitements interactifs telle la mise à jour automatique des horaires publiés.

Au chapitre 1 de ce rapport, nous discutons du cadre dans lequel s'inscrit ce stage et nous présentons le simulateur JET-SET, en spécifiant les apports de la troisième version.

Le design du système interactif-graphique de prise de décisions fait l'objet du second chapitre. Nous y discutons les problèmes rencontrés, les choix réalisés et les justifications de la restructuration du simulateur.

Le chapitre 3 présente la structure de la banque de données et le chapitre suivant décrit l'algorithme de génération des connexions servant à produire les horaires publiés. Dans ce chapitre 4, nous exposons la notion de plage horaire développée pour cet algorithme.

La réalisation proprement dite du logiciel est traitée au dernier chapitre. Nous y présentons l'implantation des différentes composantes du simulateur. C'est aussi dans ce dernier chapitre que nous discutons des outils informatiques employés et de leur adaptation à notre application.

1. SITUATION DU PROBLÈME

Au cours de ce chapitre, nous décrivons d'abord le problème de l'expérimentation et de la formation dans le domaine de la planification et la gestion du transport interurbain de passagers. Ensuite, nous présentons le simulateur d'entreprises JET-SET, dont l'objectif est d'apporter une réponse à cette problématique. Finalement, nous précisons les objectifs de la troisième version du simulateur, et en particulier ceux poursuivis au cours de ce stage.

1.1 Description de la problématique[1]

L'industrie du transport interurbain de passagers joue un rôle économique et social des plus importants. Elle constitue le support nécessaire à l'ensemble des activités professionnelles, commerciales, industrielles et sociales requérant des déplacements de personnes. Les planificateurs/gestionnaires de compagnie oeuvrant dans cette industrie doivent prendre maintes décisions touchant autant la planification et la réalisation des activités directement reliées aux services de transport offerts (choix des marchés desservis, horaires détaillés des services, itinéraires des véhicules, tarifs, ...) que la gestion des activités indirectes telles les campagnes de promotion, la requête d'études spéciales, les activités financières, etc. Cette prise de décision intervient dans un contexte de marché concurrentiel et elle doit tenir compte de l'environnement social, économique, technologique et réglementaire.

Avant de poursuivre la présentation de la problématique, précisons qu'un *service* représente ici un transport de passagers d'une ville à une autre assuré par un véhicule d'une compagnie, et que le *niveau de*

---

[1] Les sections 1.1 et 1.2, qui décrivent le cadre dans lequel ce stage a été entrepris, sont largement inspirées de Lardinois (1988).

service offert sur un marché désigne la fréquence des services sur ce marché ainsi que le tarif exigé pour chaque service. Notons finalement qu'un marché se définit comme une paire de villes (origine-destination) caractérisée par une demande de transport.

1.1.1 Environnement

La gestion d'une compagnie de transport interurbain de passagers est une tâche très complexe qui requiert une compréhension autant globale que détaillée de l'environnement dans lequel les décisions sont prises, cela d'autant plus que ces décisions sont nombreuses, diverses et relèvent de différents niveaux de direction dans l'entreprise.

Cette industrie opère sur un réseau physique (ou partiellement physique pour le cas du mode aérien). Cette dimension complexifie les mécanismes d'équilibre entre l'offre et la demande, notamment parce que certaines des caractéristiques de l'offre (temps d'attente, probabilité de trouver une place dans un véhicule, ...) dépendent non seulement des compagnies offrant les services de transports, mais aussi des réactions de la demande. De plus, la compétition entre différentes compagnies influence les niveaux de services réalisés par chacune d'elles sur chaque marché.

Dans ce contexte, il ressort que l'un des principaux problèmes d'éducation et de formation en transport, est de trouver une façon de développer chez les individus une compréhension profonde des mécanismes de compétition et d'équilibre concernés, leur permettant ainsi d'acquérir les bases de connaissances nécessaires à la planification et la gestion de réseaux de transport.

Les facteurs socioéconomiques (tendances démographiques, taux d'inflation, coût de la main-d'oeuvre, ...), technologiques (efficacité et coût de l'infrastructure et des véhicules, innovations, ...) et institutionnels (réglementation, politique de transport et de subvention, ...)

imposent par ailleurs contraintes et pressions au processus de prise de décision. Citons en exemple la hausse des prix de l'énergie pouvant obliger les gestionnaires d'une compagnie d'aviation à augmenter fortement ses tarifs, causant ainsi une baisse importante de ses parts de marché et de ses profits, le tout au profit de compagnies opérant d'autres modes de transport moins affectés par une variation des prix de l'énergie.

Une compréhension claire des impacts directs et indirects de l'environnement complexe et changeant est donc également essentielle à la bonne gestion de l'entreprise.

1.1.2 Processus de prise de décision

Tel qu'indiqué dans Lardinois (1988), la prise de décision dans une compagnie de transport interurbain de passagers se caractérise par

a) la multiplicité et la diversité des décisions à prendre;

b) la présence de plusieurs niveaux de prise de décisions et donc de plusieurs acteurs avec des objectifs différents et parfois conflictuels;

c) le degré élevé d'interdépendance et d'interaction entre tous les niveaux, tous les acteurs et toutes les décisions;

d) et l'obligation de tenir compte d'un environnement complexe et changeant.

Le processus de prise de décision est souvent présenté comme un processus réparti sur trois niveaux distincts, mais interreliés, soient les niveaux stratégique, tactique et opérationnel, qui en général correspondent respectivement à des impacts à long, moyen et court terme.

Au niveau stratégique, les décisions portent surtout sur l'établissement d'une politique d'offre, sur l'acquisition de ressources et sur les investissements majeurs. C'est à ce niveau que le planificateur s'occupe du

choix des marchés, de la configuration générale du réseau, de la planification des ressources humaines et matérielles, etc.

Les décisions tactiques portent sur l'allocation des ressources pour augmenter la productivité et la compétitivité de la compagnie. À ce niveau, il s'agit du design du réseau de services, du choix des fréquences et tarifs sur chaque marché, de l'organisation des politiques d'entretien, etc.

Au niveau opérationnel, la compagnie doit s'occuper de l'organisation détaillée, à court terme y compris quotidienne, de ses activités. C'est à ce niveau par exemple que le gestionnaire construit les routes-horaires de ses véhicules de manière à satisfaire opérationnellement les grandes lignes du cadre tactique.

1.1.3 Approche de résolution

Dans le domaine de la modélisation des transports, plusieurs modèles d'offre et de demande fournissent une aide importante aux planificateurs et aux gestionnaires. Ces modèles et les logiciels qui les intègrent ont toutefois plusieurs limites. Ils ne traitent en général que de l'un des nombreux aspects du système de transport considéré, à un niveau de décision particulier, et en faisant des hypothèses simples sur les autres composantes du système étudié. De plus, ces modèles font en général abstraction des considérations budgétaires, organisationnelles, politiques ou de marketing, et n'incluent aucunement l'élément humain dans la simulation de l'offre, perdant ainsi toute la "dimension d'incertitude" rencontrée en pratique.

Tout en admettant que les modèles mathématiques cités ci-dessus sont très utiles et jouent un rôle essentiel, il reste évident que les planificateurs et les gestionnaires ne possèdent aucun outil adéquat pour analyser et prévoir de façon complète et intégrée les effets corrélés de toutes les

décisions possibles avec l'environnement dans lequel elles sont prises. Cette constatation est d'autant plus vraie et critique dans le domaine de la formation, où on ne dispose, à toute fin pratique, d'aucun outil[1] spécifiquement dédié à la formation de planificateurs/gestionnaires et qui intègre toutes les dimensions et toute la complexité des situations rencontrées en pratique.

Pour obtenir un tel outil de formation et d'expérimentation incluant la dimension humaine, l'idée de base est de combiner des modèles de transport avec l'approche que la littérature anglosaxonne désigne par "gaming-simulation". Son objectif est de reproduire artificiellement des situations comparables avec celles que rencontrent en pratique de véritables compagnies de transport interurbain de passagers. Cette approche permet aux participants (qui jouent le rôle de planificateurs/gestionnaires en charge des compagnies) d'expérimenter progressivement les effets de leurs décisions et d'améliorer leur compréhension des mécanismes en jeu, le tout dans un environnement semblable à la réalité. Le simulateur JET-SET, présenté à la prochaine section, repose sur cette approche.

L'idée de combiner l'approche "gaming-simulation" avec des modèles de transport n'est pas nouvelle.[2] Toutefois, son application à la simulation de situations complexes impliquant différents modes de transport et plusieurs compagnies en concurrence sur différents marchés constitue une nouveauté.

Pour une analyse plus approfondie et détaillée de la problématique présentée dans cette section, le lecteur est référé à Lardinois (1988).

---

1 Il existe toutefois quelques outils, mais de portée limitée.

2 Williamsen et Ortuzar (1985) et Elias (1979) présentent deux jeux informatisés dans ce domaine.

1.2 JET-SET : un simulateur d'entreprises de transport interurbain de passagers

La conception et le développement de JET-SET a été entrepris et se poursuit au Centre de recherche sur les transports de l'Université de Montréal, sous la direction de C. Lardinois. La première version du simulateur a été complétée en 1985 et testée avec succès à l'Institut Georges A. Scott de Transports Canada, qui depuis l'utilise régulièrement dans le cadre de ses cours supérieurs en gestion des transports.[1]

Depuis, cette première version, JET-SET/1, a été distribuée dans une vingtaine d'universités et organismes au Canada, aux États-Unis et en France. Les réactions des utilisateurs ont été pour la plupart très positives, démontrant que les objectifs de formation et d'expérimentation étaient effectivement atteints.

Ces résultats encourageants ont mené à l'élaboration d'une deuxième version plus sophistiquée du simulateur qui intègre tous les éléments de la première version, ainsi que diverses améliorations provenant principalement de suggestions des utilisateurs initiaux. En particulier, cette deuxième version permet une totale déréglementation du transport, les compagnies pouvant entrer et sortir librement de chaque marché. JET-SET/2 a été complété en 1988. La suite de cette section présente sommairement cette deuxième version du simulateur.

Pendant une séance type de simulation, chaque participant (ou équipe) est en charge d'une compagnie de transport. Le maître de jeu, celui qui contrôle l'évolution du contexte où évoluent les compagnies, peut instancier différentes situations de jeu. Trois situations sont offertes : deux d'entre elles impliquent une compétition directe entre les participants sur le réseau; la troisième suppose que les participants opèrent en parallèle une même compagnie.

---

[1] Le lecteur intéressé par cette première version peut consulter Lardinois, D'Avignon, Hirou (1985).

Ces situations de jeu font intervenir différents modes de transport, soient l'autobus, le train et l'avion. Le mode automobile est aussi présent dans la simulation, mais il n'est représenté par aucun participant. Un maximum de six compagnies interagissent sur le réseau pendant une même séance de simulation.

Le réseau physique sur lequel les compagnies opèrent compte sept villes de tailles diverses. L'infrastructure routière et ferroviaire simule celle d'une région typiquement nord-américaine.

Chaque participant joue le rôle de planificateur/gestionnaire des activités de sa compagnie. À ce titre, il est appelé à prendre des décisions à chaque année[1] sur :

a) le choix des niveaux de service entre chaque paire de villes qu'il peut desservir, i.e. spécification des tarifs et fréquences journalières;

b) le choix du "niveau des services additionnels" à offrir aux passagers (i.e. transport des bagages aux terminus, repas, boissons, musique à bord des véhicules, ...);

c) les achats et locations de véhicules pour maintenir ou développer sa flotte;

d) les emprunts ou remboursements bancaires à effectuer selon la situation financière de la compagnie et les coûts et revenus projetés pour l'année suivante;

e) les campagnes de promotion, à entreprendre pour attirer les voyageurs;

f) la commande d'études spéciales (enquête origine-destination, étude de marchés, analyse des opérations de la compagnie, ...).

---

1 Une séance de simulation représente une année d'opération.

Le maître de jeu prend lui aussi plusieurs décisions au début de chaque année, notamment sur la croissance des villes, sur la variation des prix de l'énergie, des coûts de la main-d'oeuvre, des taux d'intérêts et du taux d'imposition, ainsi que sur les politiques de subvention et de réglementation.

Lorsque tous les acteurs du système ont entré leurs décisions dans le programme informatique, ce dernier "crée" l'environnement mettant en présence les compagnies et les voyageurs : il simule le comportement de ceux-ci sur le réseau multimodal et multicompagnies, et parallèlement il modélise les opérations de chaque compagnie, il ajuste l'évolution de leurs flottes de véhicules et il effectue la mise à jour de leurs situations financières.[1]

Après chaque simulation, qui correspond à une année d'activité de l'ensemble des compagnies, des rapports comportant la description de ces activités sont générés et transmis aux participants. Chaque rapport correspond à une compagnie et contient des informations sur l'ensemble du système de transport (services offerts et distribution des voyageurs sur le réseau), sur l'environnement socioéconomique et institutionnel, ainsi que sur toutes les opérations de la compagnie. Cette dernière catégorie d'informations regroupe plusieurs tableaux présentant l'état de la flotte de véhicules, les différents coûts et revenus, diverses statistiques d'opération, etc. De plus, si le participant a commandé certaines études spéciales, celles-ci sont présentes dans son rapport.

À l'aide de ce rapport, chaque participant peut analyser l'impact de ses décisions et de celles de ses concurrents, tout en tenant compte de l'évolution du contexte. Ce rapport sert de base de réflexion pour la prise de décision de l'année suivante.

---

[1] La description de ce programme de simulation suit un peu plus loin dans cette section.

Une joute complète couvre ainsi plusieurs années (séances de simulation), de sorte que l'évolution de chaque compagnie dépende non seulement des décisions du participant en charge, mais aussi des décisions des compétiteurs ainsi que de l'évolution du contexte socioéconomique et institutionnel (réglementaire), orchestré par le maître de jeu.

Le programme de simulation, comme tel, comporte quatre composantes principales. La première est un "module d'opération" qui, à partir des niveaux de services et des caractéristiques des véhicules, comptabilise les opérations physiques des compagnies et en évalue les coûts. La deuxième composante est un "module de flotte de véhicules" qui procède à la mise à jour de l'état de la flotte de chaque compagnie et qui en vérifie la compatibilité avec les niveaux de service offerts. Ce module s'occupe aussi de mettre à jour la marge de crédit bancaire, basée sur la valeur résiduelle de la flotte, et d'enregistrer les coûts des achats et locations de véhicules. Le troisième module est un "modèle de transport" possédant trois composantes. La première identifie l'ensemble des routes possibles pour chaque paire de villes et détermine leur attirance respective pour les voyageurs. La deuxième composante est un modèle de demande qui détermine la demande de transport pour chaque marché (notamment en fonction des niveaux de service offerts). Finalement, la troisième composante est un modèle de choix de routes et d'affectation des déplacements qui simule le comportement des voyageurs sur le réseau de transport. Pour une présentation détaillée de ces modèles de transport et des principes de comportements à la base de ceux-ci, le lecteur est référé à Lardinois (1987). Le dernier module du programme de simulation modélise toutes les activités financières de chacune des compagnies.

En plus du programme de simulation, le logiciel JET-SET inclut une banque de données et une interface entre l'usager et celle-ci. La banque de données regroupe un ensemble de fichiers contenant les états successifs du système de transport et des compagnies. L'interface, quant à elle, est

constituée d'un programme pour la collecte et la validation des décisions, et d'un autre pour la production des rapports à la fin de chaque séance de simulation. Finalement, le déroulement type d'une séance comporte les quatre phases successives de prise de décisions, collecte, simulation et génération des rapports.

1.3 JET-SET/3 et la gestion des opérations

L'ensemble des décisions auquel sont confrontés les utilisateurs de JET-SET/2, n'inclut aucune décision au niveau opérationnel de la gestion des compagnies. Chaque participant n'expérimente que les niveaux stratégique et tactique, sans avoir à s'occuper de la gestion détaillée des opérations de sa compagnie.

L'objectif de la troisième version du logiciel est justement de pallier à cette simplification initiale en intégrant la dimension opérationnelle, et plus précisément l'aspect "routing-scheduling", au processus d'analyse et de prise de décision : le participant devra élaborer les horaires détaillés des services qu'il offre, et donc construire les routes-horaires des véhicules de sa flotte, au lieu de simplement spécifier des fréquences journalières pour chaque marché desservi. Evidemment, cette nouvelle catégorie de décisions nécessitera la prise en compte par le participant des variations de la demande au cours d'une journée.

Une autre fonction importante du niveau opérationnel d'une compagnie est la publication des horaires publiés pour les passagers. Cette nouvelle information sera fournie automatiquement par le simulateur pour chaque compagnie, et elle sera prise en compte dans la nouvelle version du module de transport cité ci-haut. La décision de générer automatiquement les horaires publiés l'a emporté sur l'option de laisser le participant fabriquer toutes les connexions nécessaires à la formation des services indirects de ces horaires. Cette dernière option aurait été réalisable, mais plutôt longue et ardue dans le cadre d'un simulateur d'entreprises tel

JET-SET. Notre choix sous-entend donc qu'une partie du traitement de la simulation est maintenant exécutée interactivement, au fur et à mesure que le participant prend ses décisions à l'écran du micro-ordinateur.

Les conséquences de l'ajout de la dimension opérationnelle sont nombreuses, autant pour l'interface et pour la banque de données que pour le programme de simulation. En particulier, l'élaboration des horaires de service et des routes-horaires de véhicules constitue une tâche d'une complexité beaucoup plus grande que celle associée aux décisions des deux premières versions du simulateur. Ceci entraîne évidemment une modification majeure de l'interface du simulateur : il faut faire le pas d'un simple module de collecte de données à un véritable **système interactif-graphique de prise de décision**, offrant consultation et entrée de données, validation, outils d'aide à la décision et même traitement interactif des données.

Au niveau de la banque de données, une restructuration complète s'avère nécessaire en raison de l'ajout de nouvelles données très différentes et en quantités beaucoup plus volumineuse. En effet, à chaque fréquence de service conservée auparavant, correspond maintenant un horaire de service complet. De plus, l'aspect véritablement interactif du nouveau simulateur exige le développement d'une véritable banque de données plutôt que d'un simple ensemble de fichiers sans liens physiques.

Évidemment, le programme de simulation sera aussi à refaire, la majorité des modules subissant des modifications plus ou moins majeures suscitées par le niveau de détail élevé de cette troisième version de JET-SET.

1.4 Objectifs du stage

Dans le cadre d'un stage de maîtrise, on ne peut évidemment envisager le développement complet de la troisième version de JET-SET qui, à toute fin pratique, constitue un tout nouveau simulateur. Par conséquent,

l'objectif principal du stage étant l'intégration de l'aspect "routing-scheduling" au processus d'analyse et de prise de décision, nous avons développé le **système interactif-graphique de prise de décision** en nous concentrant sur les modules traitant des horaires et routes- horaires. Quant à l'élaboration de la banque de données, l'accent a été mis là aussi sur la dimension "routing-scheduling".

La nouvelle version du programme de simulation est prévue pour une phase de développement ultérieure; ainsi ce rapport de stage n'inclut aucun travail à ce niveau de simulateur.

La réalisation des objectifs cités ci-haut implique l'accomplissement des tâches suivantes.

1) La restructuration conceptuelle du processus de prise de décision à l'intérieur du simulateur.

2) La réalisation d'interfaces numériques et graphiques pour la consultation et la modification des horaires de service et des routes-horaires des véhicules. Ceci implique la création d'une représentation efficace de ces informations, pour leur manipulation par le participant.

3) La conception et l'implantation d'une banque de données pour l'entreposage et le traitement des nouvelles décisions. Toutes ces informations étant dynamiques, il faut aussi développer un ensemble de fonctions consacrées à la mise à jour de cette banque de données.

4) La réalisation du module de traitement générant automatiquement les horaires publiés offerts aux voyageurs. Ceci implique la résolution du problème de choix des connexions à effectuer pour l'obtention d'horaires publiés sensés et utiles.

1.5 Définition de termes

Nous définissons ici formellement différents termes importants utilisés dans la rédaction de ce rapport. Plusieurs autres termes plus spécifiques seront définis, au besoin, tout au long du texte.

PARTICIPANT : Individu qui participe à une séance de simulation. Il possède et gère une compagnie de transport interurbain de passagers.

USAGER : (cf. participant).

VÉHICULE : Terme général représentant un autobus, un train ou un avion.

MARCHÉ : Paire de villes (origine-destination) caractérisée par une demande de transport.

SERVICE : Transport de passagers assuré par un véhicule d'une compagnie d'une ville à une autre. Chaque service possède une origine (ville de départ), une destination (ville d'arrivée), une heure de départ et une heure d'arrivée.

SERVICE INDIRECT : Juxtaposition de deux ou trois services de manière à assurer un transport de passagers sur un marché. Plusieurs (maximum 3) véhicules peuvent se partager la réalisation d'un service indirect.

CONNEXION : Lien établi entre un service arrivant à une ville et un autre quittant cette ville plus tard. Sert à la création des services indirects.

VILLE DE TRANSFERT : Ville où s'effectue une connexion à l'intérieur d'un service indirect. Parfois appelé "transfert" pour simplifier.

ACTIVITÉ : Correspond à une arrivée ou à un départ de service en une ville.

ITINÉRAIRE : Suite de villes décrivant tout trajet possible des voyageurs d'un marché donné, c'est-à-dire d'une origine à une destination.

HORAIRE DE SERVICE : Liste de services en ordre chronologique de départ entre deux villes.

HORAIRE PUBLIÉ : Liste de services et de services indirects annoncés aux voyageurs en ordre chronologique de départ pour un marché. Cet horaire est généré à partir des horaires de service.

HORAIRE : Ensemble des horaires de service et des horaires publiés d'une compagnie.

ROUTE-HORAIRE : Suite chronologique de services qu'accomplit un véhicule dans une journée. Parfois appelé "route" pour simplifier.

2. DESIGN DU SIMULATEUR

Au cours de ce chapitre, nous décrivons le simulateur d'entreprises JET-SET/3. Cette description porte sur les interfaces du système interactif-graphique de prise de décision intégré au simulateur. Dans un premier temps, nous présentons la nouvelle organisation générale des interfaces, en décrivant la "structure départementale" qui définit l'environnement de prise de décision auquel font face les planificateurs/gestionnaires lors d'une séance de simulation. Pour compléter cette première partie, nous exposons le déroulement type d'une séance de simulation.

Dans un deuxième temps, nous donnons le détail des interfaces développées principalement pour les départements traitant de l'aspect "routing-scheduling".

2.1 Structure départementale

Les décisions à prendre dans JET-SET/2 étaient relativement simples, statiques (i.e. en quantité fixe) et limitées en nombre. La complexité de la tâche provenait surtout de leurs interactions à l'intérieur d'un environnement complexe et compétitif.

La construction des horaires de service dans JET-SET/3 nécessite au contraire un nombre indéterminé, mais en général important de décisions plus complexes. Le participant ajoute, enlève et modifie ses services jusqu'à obtenir les horaires de service appropriés. De plus, l'insertion d'un service entre une paire de villes affecte le marché correspondant à cette paire de villes, mais aussi plusieurs autres marchés qui seront desservis indirectement par ce service. Ainsi la consultation régulière des horaires publiés sur l'ensemble des marchés s'avère-t-elle nécessaire à la prise de décision.

L'élaboration des routes-horaires des véhicules nécessite encore plus de choix, de manipulations et d'ajustements, que ce soit pour satisfaire les directives tactiques ou pour sauver quelques véhicules.

Ces observations ont mené à la conception d'un véritable système interactif-graphique de prise de décision, où l'usager a constamment accès aux informations pertinentes, ainsi qu'à un ensemble d'outils pour l'aider dans la réalisation de ses horaires et routes-horaires.

Ce système de prise de décision est interactif et dynamique afin de permettre l'analyse et la prise de décision à l'écran. La liste suivante présente les diverses décisions avec lesquelles le système doit composer :

1) l'horaire de service pour chaque paire de villes;
2) la route-horaire de chaque véhicule;
3) la tarification;
4) le niveau de services additionnels offerts aux voyageurs;
5) la réalisation de campagne de promotion;
6) la réalisation d'études spéciales;
7) les achats et/ou locations de véhicules;
8) les emprunts et/ou remboursements bancaires.

Ces décisions sont très diverses : certaines ont trait à la planification, d'autres au marketing, d'autres aux finances, etc. Dans la réalité, ces décisions sont normalement prises par une équipe de personnes, chacune ayant son domaine de spécialisation. De plus, un niveau élevé de communication doit régner dans l'organisation étant donné le degré élevé d'interdépendance et d'interaction entre toutes les décisions.

Par conséquent, nous avons décidé de concevoir un système qui répartit les décisions selon leur domaine d'application et qui permet le transfert d'informations entre les *départements* correspondants. L'idée de

base est de permettre la prise de décision dans un département tout en tenant compte des informations et attentes des autres départements.

Ainsi le système interactif de prise de décision présente six départements décisionnels :

1) le département Planification;
2) le département Gestion des opérations;
3) le département Marketing;
4) le département Achat et location;
5) le département Finances;
6) le département Statistiques et études spéciales.

Le **département Planification,** en charge des politiques et stratégies générales de la compagnie, a pour responsabilité le choix des marchés à desservir et des niveaux de services offerts. C'est donc ce département qui décide des horaires publiés que la compagnie désire annoncer aux voyageurs sur les différents marchés qu'il choisit de desservir. Ces horaires publiés doivent tenir compte des parts de marchés visées, des implications financières et de la réglementation (si elle est présente). Cependant, ils ne constituent que des recommandations, des cibles élaborées du seul point de vue du responsable de la planification, sans considération de faisabilité.

Ces horaires publiés ne sont pas construits de façon directe, mais ils sont générés automatiquement à partir des horaires de service. C'est sur ces derniers que portent en réalité les décisions du département : pour chaque paire de villes qu'il veut desservir de façon directe, il construit un horaire de service qui définit tous les services proposés sur cette paire pour une journée type. De ces horaires de service sont générés les services indirects desservant chaque marché, et donc les horaires publiés. Le terme "horaire cible" est utilisé pour désigner l'ensemble des horaires de service et des horaires publiés.

Le **département Gestion des opérations** a pour responsabilité la construction des routes-horaires des véhicules. C'est ce département qui en fait détermine les horaires de service véritablement assurés par les véhicules de la compagnie et donc les horaires publiés qui seront annoncés aux voyageurs. Le terme "horaire opérationnel" est utilisé pour désigner l'ensemble des horaires de service et des horaires publiés générés automatiquement à partir des routes-horaires.

En construisant les routes-horaires de ses véhicules, le département doit s'assurer que l'horaire opérationnel qui en résulte correspond au mieux à l'horaire cible du département Planification.

Le **département Marketing** établit le prix des billets. Ces tarifs sont spécifiés pour chaque itinéraire (considéré dans l'horaire publié) de chaque marché. Ce département détermine également les niveaux de services additionnels offerts aux passagers et commande les campagnes de promotion et études spéciales.

Les décisions sur l'achat ou la location de véhicules pour satisfaire les routes-horaires et les niveaux de service envisagés pour l'année à venir (et éventuellement les suivantes), se prennent au **département Achat et location.** Les coûts de ces achats et locations peuvent entraîner une révision des niveaux de service offerts sur l'ensemble du réseau.

Les emprunts et remboursements bancaires sont décidés au **département Finances,** et tiennent compte de la situation financière de la compagnie, des modifications prévues aux opérations et des achats ou locations de véhicules. L'analyse de ces différents facteurs peut entraîner ici aussi une révision des décisions prises dans les autres départements.

Finalement, le **département Statistiques et études spéciales** dispose des résultats détaillés de la dernière séance de simulation ainsi que de toutes les études spéciales commandées par le participant. Aucune décision n'est prise dans ce département, les participants n'y accèdent qu'à des fins de consultation.

2.2 Déroulement d'une séance de simulation

Une séance type de simulation avec JET-SET/3 comporte trois phases, soient :

1) la prise de décision interactive;
2) la simulation proprement dite de l'année;
3) la génération des rapports annuels.

Ces trois phases sont réitérées successivement année après année. La première phase nécessite plusieurs tâches qui s'effectuent selon un certain scénario. Celui présenté ci-dessous présuppose pour chaque compagnie une équipe de participants plutôt qu'un seul. Cinq personnes peuvent être en charge respectivement des cinq départements décisionnels, mais pour une répartition plus logique des décisions, il est préférable d'utiliser trois personnes : un directeur de la planification et du marketing, un directeur des opérations et un directeur pour les achats/locations et les finances. Cependant rien n'exclut qu'une seule et même personne soit en charge de tous les départements. La présentation suivante du déroulement type d'une séance de simulation décrit le rôle de chaque directeur de département, peu importe le nombre de personnes effectivement en charge de la compagnie.

Dans un premier temps, le "directeur de la planification" définit les nouveaux horaires de service pour l'année qui vient. Pour ce faire, il consulte les résultats et statistiques de l'année précédente pour sa compagnie et les autres, et il effectue les modifications appropriées. Le produit final est un horaire cible.

Dans un deuxième temps, le "directeur des opérations" définit les routes-horaires de ses véhicules en essayant de satisfaire au maximum l'horaire cible du département Planification. Pour ce faire, il consulte l'horaire cible et le compare à l'horaire opérationnel généré par ses

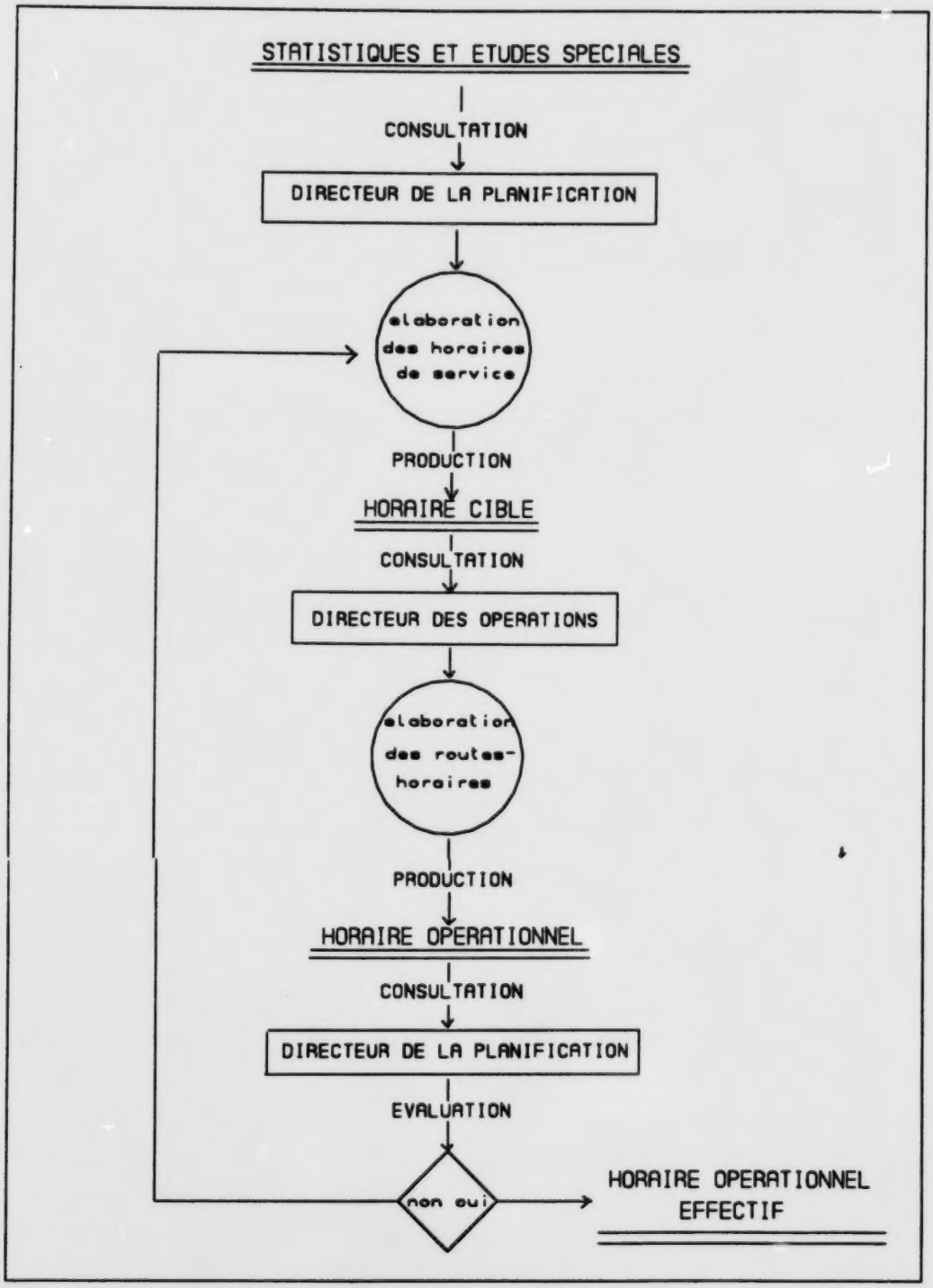


Figure 2.1 : Processus d'élaboration de l'horaire opérationnel effectif

routes-horaires. Le participant s'efforce d'obtenir un "routing" des plus efficaces et qui de plus soit réalisable, notamment au niveau de l'équilibre entre les véhicules situés aux différentes villes en début de journée et à la fin de la journée. La construction de ces routes-horaires s'effectue à l'aide d'une interface graphique permettant une manipulation aisée des routes. Le participant bénéficie aussi d'un ensemble d'outils pour l'aider dans sa conception. S'il le juge opportun, le directeur des opérations peut "négocier" avec le directeur de la planification afin d'amener celui-ci à modifier son horaire cible.

Le directeur des opérations termine la construction de l'horaire opérationnel lorsqu'il considère s'être approché au mieux de l'horaire cible. Le directeur de la planification consulte alors cet horaire opérationnel et peut proposer d'autres changements. Le directeur des opérations ajuste ses routes-horaires en conséquence; et ainsi de suite jusqu'à entente commune (cf. figure 2.1).

À la fin de cette étape l'horaire opérationnel devient effectif[1] et le "directeur du marketing" y inclut les tarifs. Ceux-ci sont spécifiés pour chaque itinéraire de chaque marché. Ce directeur décide aussi des niveaux de services additionnels offerts aux passagers. C'est également lui qui décide de la nécessité d'effectuer une campagne de promotion ou de commander des études spéciales.

Par la suite le "directeur des achats et locations" évalue le nombre de véhicules à acheter et/ou louer, étant donné l'état de sa flotte et les décisions prises au niveau du "routing". Il peut aussi "négocier" avec les directeurs de la planification et des opérations de même qu'avec le directeur des finances s'il s'avère que le nombre de véhicules à acheter ou louer est trop important.

---

[1] C'est-à-dire que cet horaire sera effectivement utilisé à la simulation.

Finalement, le "directeur des finances" effectue les emprunts ou remboursements bancaires qu'il juge justifiés par les activités de la compagnie, mais il peut lui aussi "négocier" avec les autres départements dont les décisions ont des impacts financiers qu'il juge trop important.

Lorsque toutes les décisions sont entrées et validées, le simulateur est activé sur l'ensemble des données du système. À la fin de la simulation, les rapports des compagnies sont générés et les participants y accèdent par le département Statistiques et études spéciales.

Tel que suggéré dans la description ci-dessus d'un scénario type, la communication verbale à l'intérieur d'une équipe de participants, en charge d'une compagnie, constitue un aspect important et original de JET-SET/3. Cette communication n'est toutefois pas indispensable; les participants peuvent ignorer leurs "collègues", mais en risquant que la stratégie globale ne satisfasse aucun d'eux, notamment du point de vue des résultats financiers. Une seule personne peut aussi être en charge des cinq départements décisionnels. De plus, l'étape de planification n'est pas indispensable. Si elle n'est pas réalisée, l'horaire cible est inexistant et l'horaire opérationnel généré à partir des routes-horaires est automatiquement considéré pour la simulation. En résumé, le simulateur permet à un seul ou à plusieurs individus d'expérimenter toutes les facettes de la gestion d'une compagnie de transport.

2.3 Description des interfaces

Deux départements traitent de l'aspect "routing-scheduling" soient celui de la Planification et celui de la Gestion des opérations. Ces deux départements font l'objet des sections suivantes.

2.3.1 Département Planification

L'interface développée pour ce département doit permettre :

1) la construction et la consultation des horaires de service pour l'année à venir;

2) la consultation de différents horaires (ex. : l'horaire opérationnel du département Gestion des opérations, l'horaire effectif de l'année précédente, l'horaire effectif de l'année précédente d'une autre compagnie, ...);

3) un accès au département Statistiques et études spéciales pour consultation pendant la prise de décision;

4) la spécification de villes de transfert supplémentaires à ceux automatiquement considérées lors de la génération des services indirects des horaires publiés. Ces villes de transfert additionnelles seront aussi considérées pour les horaires publiés du département Gestion des opérations.

La liste qui suit définit les différentes composantes de l'interface illustrée à la figure 2.2.

Élément	Description
Zone de consultation	Partie de l'écran où le participant consulte les informations nécessaires à sa prise de décision.
Zone de travail	Partie de l'écran où le participant construit l'horaire cible.
Ligne de message	Dernière ligne de l'écran contenant tous les messages et directives envoyés à l'usager.
Ligne titre	Ligne apparaissant en haut de chacune des zones et spécifiant l'information contenue dans celle-ci.
Ligne de commande	Partie d'une zone présentant les commandes disponibles pour la manipulation des données.
Fenêtre de service	Partie d'une zone contenant un horaire de service entre deux villes.
Fenêtre d'horaire publié	Partie d'une zone contenant l'horaire publié d'un marché.

Élément	Description (suite)
Case origine	Partie d'une fenêtre de service ou d'horaire publié qui contient la ville de départ des services ou services indirects considérés dans la fenêtre.
Case destination	Partie d'une fenêtre de service ou d'horaire publié qui contient la ville d'arrivée des services ou services indirects considérés dans la fenêtre.

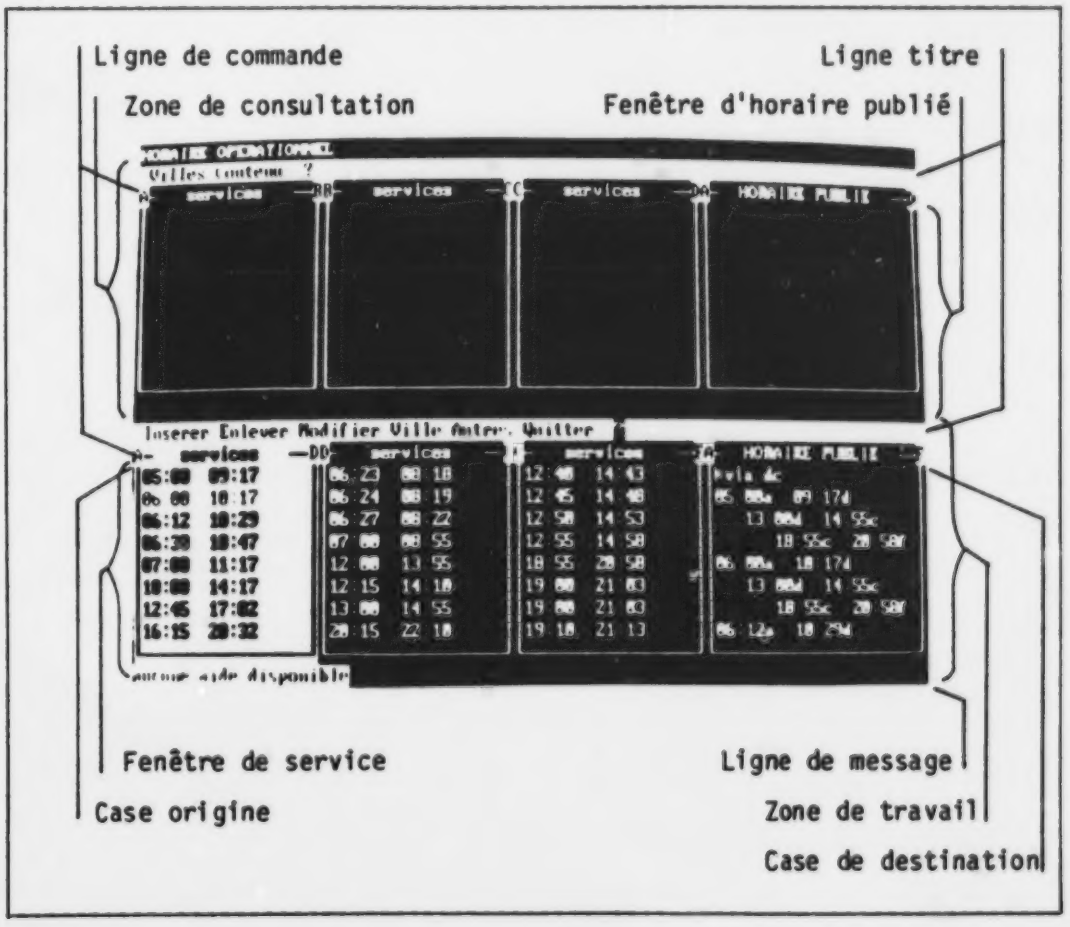


Figure 2.2 : Interface du département Planification

La configuration utilisée pour présenter un horaire, que ce soit pour la consultation ou la construction, comporte quatre fenêtres : trois pour des horaires de service et une pour un horaire publié. Cet horaire publié est, rappelons-le, généré automatiquement à partir des horaires de service. Étant donné que les itinéraires considérés pour les services indirects d'un marché ont au plus deux transferts, et donc empruntent au plus trois services, la juxtaposition de trois fenêtres de service permet d'étudier les connexions ayant menées aux services indirects de l'horaire publié de ce marché. Par exemple, le participant peut dans la première fenêtre afficher l'horaire de service de OTTAWA-MONTRÉAL, dans la deuxième fenêtre celui de MONTRÉAL-TROIS-RIVIÈRES et dans la dernière fenêtre de service afficher celui de TROIS-RIVIÈRES-QUÉBEC. Le participant peut ainsi examiner toutes les connexions menant aux services indirects de l'horaire publié du marché OTTAWA-QUÉBEC qu'il peut charger dans la fenêtre d'horaire publié.

2.3.1.1 Utilisation générale

Cette section offre une brève description des fonctions offertes pour opérer l'interface du département Planification. Ces fonctions n'incluent aucunement les commandes de manipulation des données qui, elles, seront présentées à la section suivante.

2.3.1.1.1 Activer une zone

Pour opérer une zone – celle de consultation ou celle de travail – l'usager doit s'assurer de l'activité de celle-ci. L'activité d'une zone se définit comme un état permettant l'accès à toutes les commandes de manipulation des données de la zone ainsi qu'aux fonctions d'opération générale sur celle-ci. Une seule zone peut être active à la fois, lorsque la zone de

travail est active, celle de consultation est inactive, c'est-à-dire qu'aucune opération ou manipulation ne peut y être effectuée; et inversement si la zone de consultation est active.

Pour interchanger l'état (actif/inactif) des zones, l'usager utilise la touche <F1>.

2.3.1.2 Changer la fenêtre courante

Dans une zone active, la fenêtre courante est affichée d'une couleur plus claire soulignant son état actif. Les opérations effectuées par l'usager portent toujours sur la fenêtre courante. Celle-ci peut être une fenêtre de service ou une fenêtre d'horaire publié. Pour changer de fenêtre courante à l'intérieur d'une zone active, on utilise les touches de direction ou la souris.

2.3.1.1.3 Choisir une commande

Pour choisir une commande offerte dans la ligne de commande de la zone active, le participant peut taper la première lettre de la commande ou encore pointer sur la commande avec la souris. Si un menu d'options est associé à cette commande, il apparaît automatiquement et le participant fait son choix dans ce menu selon le même procédé.

2.3.1.1.4 Remplir une fenêtre de données

Lorsque de l'information additionnelle est requise pour effectuer une opération, le système ouvre une fenêtre où le participant insère les données nécessaires. Si cette fenêtre possède plusieurs champs d'informations à compléter, l'usager peut passer d'un champ à l'autre en utilisant les touches de direction ou la souris.

2.3.1.1.5 Défiler une liste

Les horaires affichés dans les différentes fenêtres de l'interface peuvent être défilés ("scrollés") par l'usager. Ce défilement est continu et s'effectue avec les touches de directions habituelles, soient ↑, ↓, PgUp, PgDn.

2.3.1.1.6 Aide

Aucune aide interactive n'est encore offerte, mais cette option est prévu dans le système final. Elle sera accessible par le symbole ? de la ligne de commande de la zone active.

2.3.1.1.7 Quitter

Pour quitter le département, le participant utilise l'option *QUITTER* de la ligne de commande ou la touche < ESC > .

2.3.1.2 Manipulation des données

2.3.1.2.1 Zone de consultation

Lorsque la zone de consultation est active, l'usager a accès à trois opérations offertes par la ligne de commande. La commande *VILLE* permet de remplacer l'horaire de service ou publié présenté dans la fenêtre courante par l'horaire de service d'une autre paire de villes ou par l'horaire publié d'un autre marché, selon le cas. Pour ce faire, l'usager spécifie directement dans les cases origine et destination les nouvelles villes considérées.

La commande *CONTENU* permet de changer l'information courante de la zone de consultation pour un autre type d'information. Un sous-menu présente les informations disponibles, soient l'horaire opérationnel du département Gestion des opérations, l'horaire effectif de l'année précédente, l'horaire effectif de l'année précédente d'une autre compagnie ou un accès au département Statistiques et études spéciales.

2.3.1.2.2 Zone de travail

Lorsque la zone de travail est active, la ligne de commande présente sept commandes de manipulation : *INSERER*, *ENLEVER*, *MODIFIER*, *VILLE*, *AUTRES*, *QUITTER* et *?*.

Les options *VILLE* et *?* ont le même effet que pour la zone de consultation (cf. section 2.3.1.2.1). La commande *QUITTER*, quant à elle, permet, tel que précisé plus haut, de sortir du département.

Les autres options permettent de manipuler les services des horaires. La commande *INSERER* permet d'ajouter un nouveau service dans l'horaire de service de la fenêtre courante. Pour cela, l'usager spécifie l'heure de départ du service dans une fenêtre de données; l'heure d'arrivée est calculée par le système. Suite à cette opération, la banque de données est mise à jour instantanément; le contenu des fenêtres reflète le nouvel état de la banque de données.

Pour enlever un service de l'horaire de service de la fenêtre courante, l'usager se positionne sur ce service et choisit la commande *ENLEVER*. Une mise à jour de la banque de données et des fenêtres est aussitôt effectuée.

La commande *MODIFIER* permet au participant de déplacer l'heure de départ d'un service. Le participant n'a qu'à spécifier cette nouvelle heure dans la fenêtre de données fournie par le système.

La commande *AUTRES* présente un menu de fonctions indirectement reliées à la construction de l'horaire, comme par exemple une fonction permettant de spécifier des transferts additionnels à considérer pour la génération automatique des services indirects des horaires publiés.

Au terme du stage, la majorité des fonctions du département Planification sont opérationnelles. Seuls restent à implanter les utilitaires du menu de la commande *AUTRES*, ainsi que la fonction pour changer le contenu de la zone de consultation.

2.3.2 Département Gestion des opérations

L'interface développée pour ce département doit permettre :

1) la construction et la consultation des routes-horaires des véhicules;
2) la consultation de l'horaire opérationnel généré automatiquement à partir des routes-horaires;
3) la consultation de l'horaire cible du département Planification.

2.3.2.1 Représentation graphique

Une route-horaire de véhicule se définit comme une suite chronologique de services effectués par un seul véhicule à l'intérieur d'une journée de travail. Nous avons recherché une représentation de ces routes-horaires qui puisse aider le participant dans leur construction.

La représentation la plus simple et spontanée est celle dérivée de la définition, soit une juxtaposition de services en ordre chronologique.

Véhicule	Route-horaire		
18	A, 09:00	B, 11:00	E, 14:00
	B, 10:30	D, 12:58	F, 14:53
19	C, 07:00	F, 10:58	
	F, 11:10	G, 13:55	

où chaque service correspond à un quadruplet

$$\begin{pmatrix} \text{ville de départ, heure de départ} \\ \text{ville d'arrivée, heure d'arrivée} \end{pmatrix}.$$

Cette représentation est simple et on peut y voir les conflits possibles (ex. : le départ trop tôt du deuxième service de la route du véhicule 19). Toutefois rien n'est mis en évidence. Il faut lire en détail les suites de services représentant les routes pour s'apercevoir des conflits ou simplement pour décider des manipulations avantageuses. Bref, la tâche du gestionnaire n'est nullement facilitée par cette représentation des routes-horaires.

Après avoir envisagé d'autre représentations, nous avons finalement choisi une représentation qui exprime une route-horaire par un tracé dans un diagramme véhicule-temps (cf. figure 2.3)[1].

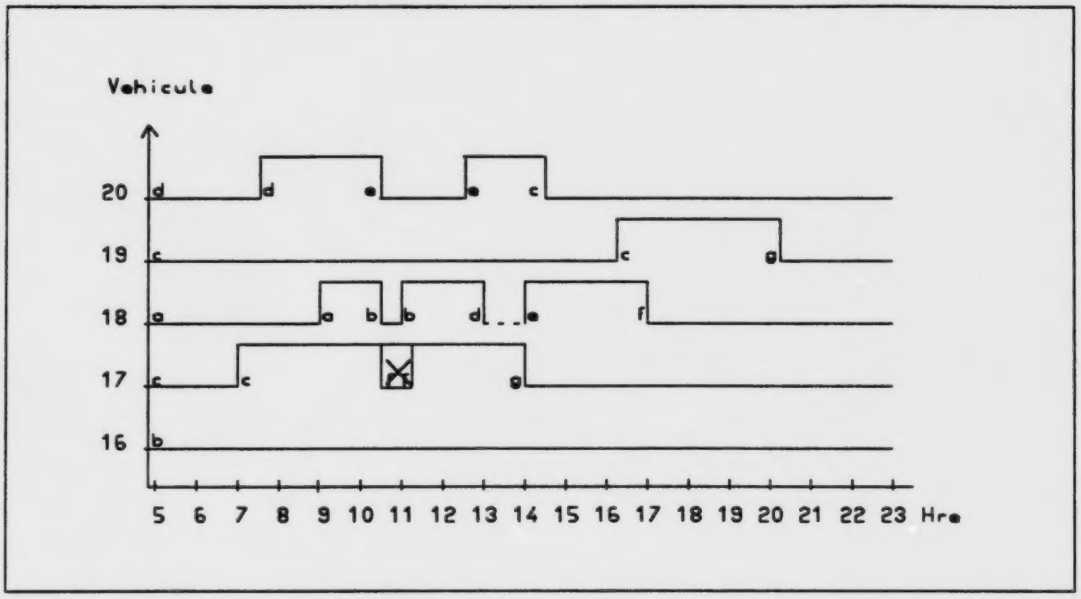


Figure 2.3 : Diagramme véhicule-temps de routes-horaires

L'axe horizontal du diagramme indique les heures d'une journée de travail, que l'on suppose étalée de 5 hres à 23 hres, et l'axe vertical identifie tous les véhicules de la compagnie. Une route-horaire est représentée par une suite d'oscillations dans le temps. Le creux d'une oscillation indique que le véhicule est en attente dans une ville, tandis que le haut de l'oscillation représente un déplacement entre deux villes, c'est-à-dire un service. L'origine et la destination du service effectué sont inscrites dans les extrémités de l'oscillation.

---

1 L'idée de base de cette représentation s'inspire des graphiques représentant les tâches des véhicules que consultent les compagnies aériennes.

Prenons par exemple le véhicule #20 de la figure 2.3, sa route-horaire se présente comme suit :

> *"Le véhicule est localisé à la ville D. À 7h30*[1] *il effectue un service vers la ville E, qu'il atteint à 10h20. Le véhicule est en attente jusqu'à 12h40, puis il accomplit un service vers la ville C. Il reste en C de 14h25 à la fin de la journée, soit 23 hres."*

Cette représentation graphique a l'avantage de permettre d'un seul coup d'oeil, l'examen de la réalisabilité et de l'efficacité de plusieurs routes-horaires. L'usager peut distinguer rapidement les manipulations avantageuses à réaliser. Par exemple, de la figure 2.3, on voit immédiatement que le transfert du service c-g du véhicule #19 au véhicule #20 est réalisable et efficace : il permet de sauver un véhicule.

Ce diagramme permet également de mettre facilement en évidence les conflits existants dans une route-horaire. Notamment l'utilisation d'une ligne pointillée au lieu d'une ligne pleine entre deux services dénote un conflit de villes, c'est-à-dire que la ville de départ d'un service ne correspond pas à la ville d'arrivée du service précédent. Les conflits de temps, quant à eux, ressortent d'eux-même puisque si l'heure de départ d'un service précède l'heure d'arrivée du service précédent, les oscillations correspondantes se chevauchent. Pour faire ressortir davantage l'invalidité d'une telle route, le système marque d'un *X* tout conflit de temps.

---

[1] Le lecteur attentif remarquera qu'il est difficile de déterminer sur le diagramme l'heure exacte d'un départ ou d'une arrivée de service, toutefois on verra plus loin comment l'usager peut déterminer aisément celles-ci.

Cette représentation simple et efficace a de plus l'avantage de se manipuler aisément. La prochaine section précise comment.

2.3.2.2 Présentation des interfaces

Le participant a le choix entre deux interfaces pour construire ses routes-horaires. L'interface 1 intègre le diagramme des routes-horaires, les commandes de manipulation et d'opération, ainsi que l'accès à des outils d'aide à la décision. L'interface 2 comprend en plus de ces composantes, une zone de consultation de l'horaire opérationnel généré à partir des routes-horaires. Cette zone occupe la moitié supérieure de l'écran, et confine ainsi le diagramme dans la partie du bas. Dans cette interface le diagramme présente cinq routes-horaires à la fois, comparativement à dix pour l'interface 1.

En général, le participant utilisera les deux interfaces. L'interface 2 est en particulier utile pour vérifier la mesure dans laquelle l'horaire opérationnel, généré par les routes-horaires, satisfait l'horaire cible du département Planification. L'interface 1, quant à elle, par sa plus grande surface d'affichage de routes, permet de travailler au niveau de l'efficacité du "routing" global.

Nous allons tout d'abord présenter l'interface 1, puis les composantes supplémentaires de l'interface 2.

2.3.2.2.1 Interface 1

La liste ci-dessous décrit les différentes composantes de l'interface 1 (cf. figure 2.4).

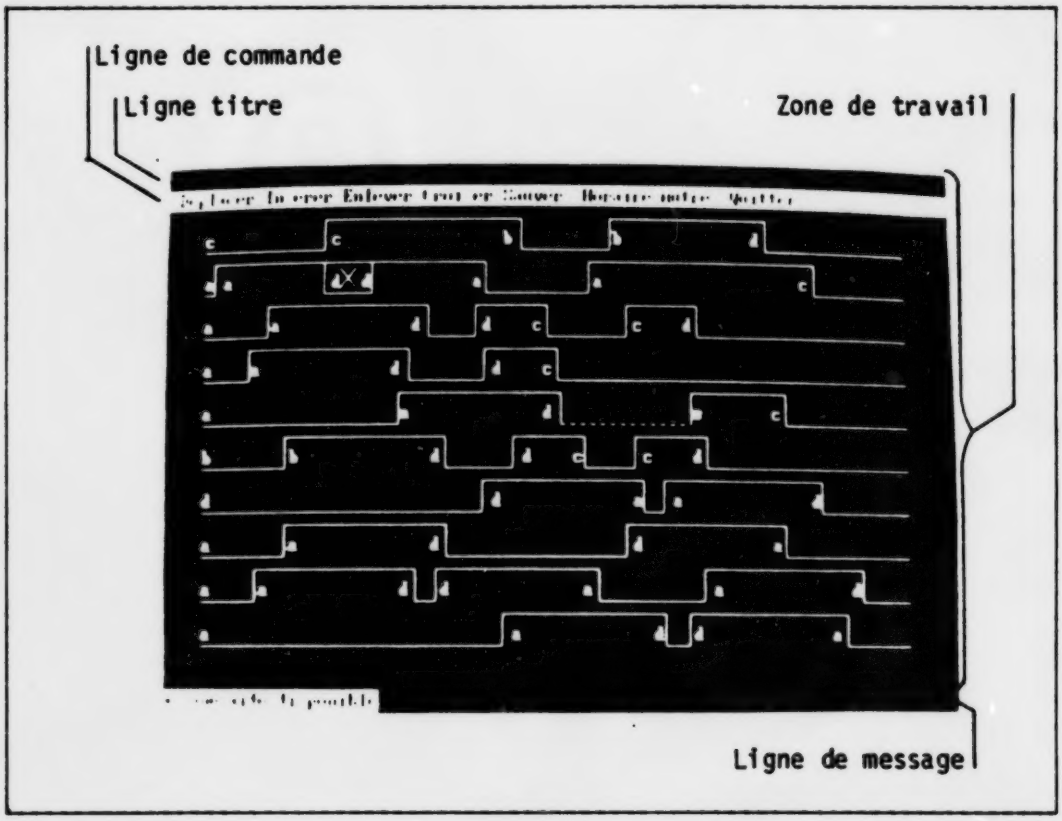


Figure 2.4 : Interface 1 du département Gestion des opérations

Élément	Description
Zone de travail	Partie de l'écran où le participant construit ses routes-horaires.
Ligne titre	Ligne située dans le haut de la zone de travail et spécifiant l'information contenue dans celle-ci.
Ligne de commande	Partie de la zone de travail présentant les commandes disponibles pour manipuler les informations de la zone.
Ligne de message	Dernière ligne de l'écran contenant tous les messages et directives envoyés à l'usager.

Le diagramme des routes-horaires occupe toute la surface de la zone de travail. L'axe vertical peut être défilé et présente dix véhicules à la fois. L'axe horizontal est fixe et représente une journée de travail, soit de 5 hres à 23 hres.

Le défilement des véhicules sur l'axe vertical est continu et s'effectue avec les touches de direction habituelles. Notons aussi que le procédé pour activer une commande est le même que celui décrit pour l'interface du département Planification (cf. section 2.3.1.1.3).

L'axe horizontal du graphique étant gradué en heures, il est difficile de déterminer visuellement l'heure exacte de départ ou d'arrivée d'un service. Pour pallier à cet inconvénient, nous avons intégré une "fonction d'inspection" qui détaille le service pointé à l'écran. Ainsi lorsqu'un participant pointe avec la souris un service d'une route-horaire, une fenêtre s'ouvre sur celui-ci et présente ses coordonnées espace-temps. Le participant pourrait éventuellement avoir accès à d'autres types d'information (ex. : le tarif, le niveau de services additionnels, la durée du voyage, etc.) en utilisant un bouton différent de la souris.

Plusieurs commandes de manipulation des routes nécessitent de déterminer une *section de route* sur laquelle on veut faire porter l'opération. Cette sélection s'effectue en pointant avec la souris sur le premier et le dernier service de la section de route. Évidemment, si la sélection ne porte que sur un service, il faut pointer deux fois sur celui-ci.

Quatre commandes sont disponibles pour modifier les routes-horaires, soient *DEPLACER, INSERER, ENLEVER* et *CROISER*.

La commande *DEPLACER* permet à l'usager de déplacer l'heure de départ d'un service ou d'une suite de services. Après avoir activé la commande, le participant sélectionne la section de route à déplacer, puis inscrit la nouvelle heure de départ dans la fenêtre envoyée par le système. Cette nouvelle heure agit sur l'ensemble des services de la section de route, ceux-ci sont déplacés en conséquence. La mise à jour de la banque de données est effectuée et la route-horaire est réaffichée.

La commande *INSERER* présente deux options, soient l'insertion manuelle ou l'insertion venant d'une copie de route. Dans le premier cas, l'usager fournit les données nécessaires dans une fenêtre envoyée par le système, et la mise à jour s'effectue. Pour l'insertion par copie, l'usager doit sélectionner une section de route à copier et ensuite spécifier un point d'insertion dans une autre route-horaire. La mise à jour de la banque de données s'effectue aussitôt et la route est réaffichée en conséquence.

Pour éliminer une section de route, le participant active la commande *ENLEVER* puis sélectionne la section de route à enlever. La mise à jour s'effectue et la route est réaffichée.

Le participant peut également croiser (ou interchanger) des sections de route. Pour cela, il utilise l'option *CROISER* et indique avec la souris les deux points de croisement. Ces points, afin d'être valides, doivent correspondre à une attente à la même ville des deux véhicules concernés. Le système effectue le croisement tel qu'illustré à la figure 2.5.

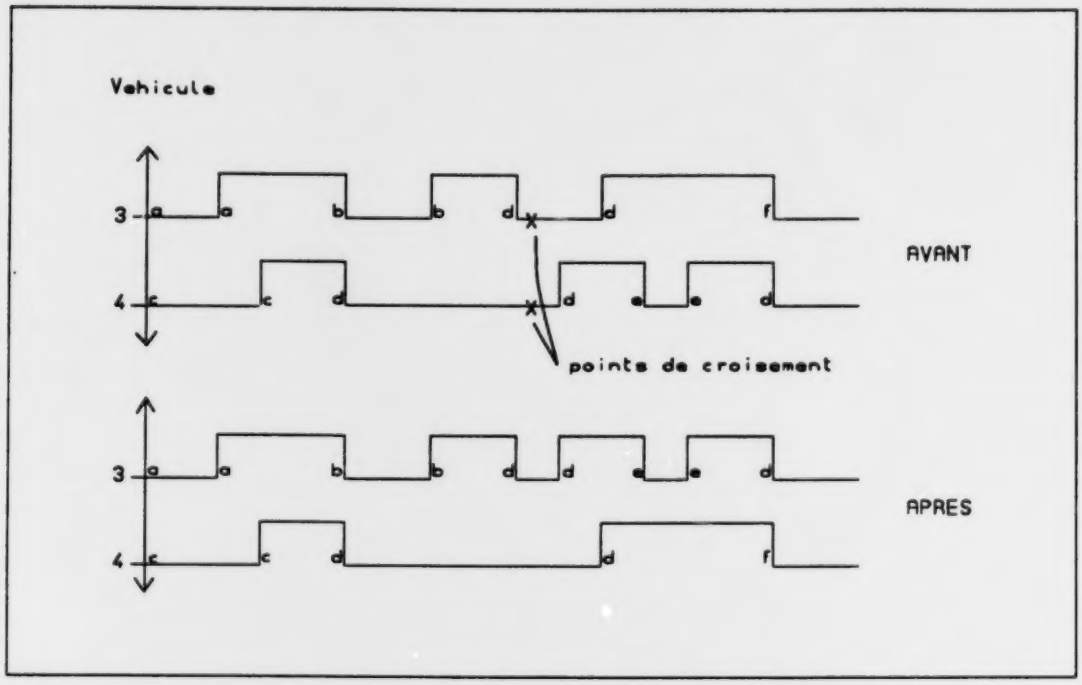


Figure 2.5 : Croisement de routes-horaires

Ces quatre commandes (*DEPLACER, ENLEVER, INSERER et CROISER*) suffisent pour manipuler facilement les routes-horaires. Les autres commandes disponibles sont des utilitaires indirectement reliés à la manipulation des routes. Parmi celles-ci, l'option *SAUVER* permet de sauvegarder sur disque la banque de données située en mémoire (cette opération s'effectue aussi automatiquement lorsque le participant quitte le système). L'option *HORAIRE* sert à passer à l'interface 2 du département (présentée un peu plus loin) et la commande *AUTRES* permet d'accéder à un ensemble d'outils d'aide à la décision. Celui-ci inclut pour l'instant une

fonction pour changer l'ordre des véhicules sur l'axe vertical, une fonction de validation des routes-horaires en fonction de l'infrastructure et de la réglementation, ainsi qu'un accès au diagramme espace-temps des routes-horaires des véhicules affichés.

Ce dernier outil est une interface de consultation (cf. figure 2.6) qui présente les routes-horaires sous une autre forme que celle du graphique véhicule-temps présenté plus haut.

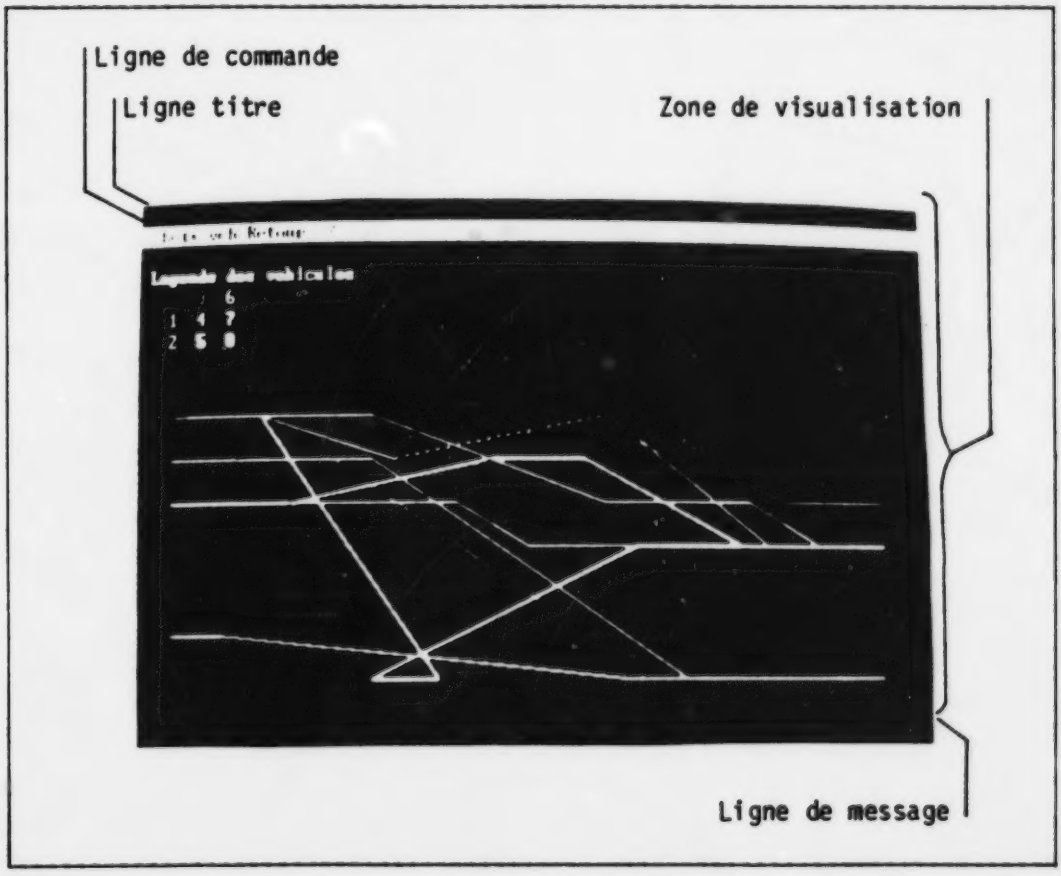


Figure 2.6 : Interface de consultation

Cet écran est divisé en quatre parties, soient une zone de visualisation, une ligne titre, une ligne de commande et une ligne de message. La zone de visualisation présente un diagramme espace-temps de routes-horaires. L'axe vertical représente l'ensemble des villes et l'axe horizontal est le même que celui du diagramme véhicule-temps. Chaque route-horaire est tracée dans l'espace et le temps : un trait horizontal indique un temps d'attente dans une ville (l'ordonnée), tandis qu'une ligne oblique désigne un déplacement entre deux villes. Une ligne pointillée indique un conflit de ville entre deux services successifs assurés par un même véhicule. Chaque route est représentée par une couleur différente; le numéro de véhicule correspondant à chacune est indiquée dans la légende fournie avec le diagramme.

Si plusieurs routes-horaires possèdent des sections de route en commun, celles-ci se superposent et seulement la couleur de la dernière route-horaire affichée est visible. Pour pallier à cet inconvénient, le participant peut pointer le numéro du véhicule de la route-horaire désirée dans la légende et celle-ci sera retracée.

Ce diagramme espace-temps est très utile pour le participant, car il permet de souligner les sections de route-horaire en commun entre les véhicules, et ainsi de suggérer certaines manipulations. Plus particulièrement, il permet à l'usager de savoir rapidement quels véhicules sont disponibles en une ville à une heure donnée.

Les commandes disponibles avec cette interface de consultation sont *CHOIX-VEH*, *RETOUR* et *?*. La première option permet de modifier le choix des routes de véhicules affichées. Le *?* sert, ici aussi, d'accès au système interactif d'aide. Finalement, le retour à l'interface du département (celle utilisée au moment de l'appel à cet outil, soit 1 ou 2) se réalise par la commande *RETOUR*.

Pour quitter le département Gestion des opérations, le participant a le choix entre la commande *QUITTER* ou la touche < ESC > .

2.3.2.2.2 Interface 2

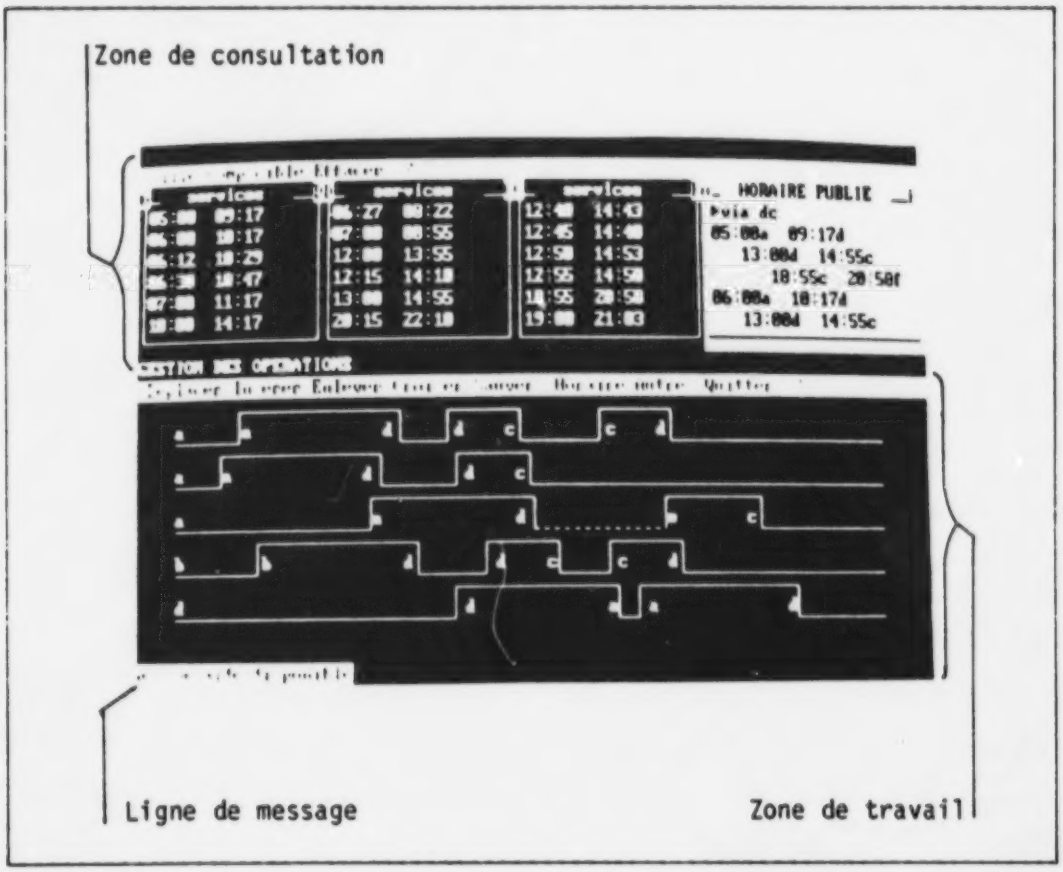


Figure 2.7 : Interface 2 du département Gestion des opérations

Cette interface fonctionne selon la même principe que celle du département Planification : une seule des zones est active à la fois, et pour changer de zone active, on utilise la touche <F1> .

L'horaire opérationnel est présenté sous la même configuration que dans le département Planification et se manipule de la même façon (cf. sections 2.3.1.1 et 2.3.1.2).

Les commandes disponibles, lorsque la zone de consultation est active, sont *VILLE*, *COMP-CIBLE*, *EFFACER* et *?*. Les commandes *VILLE* et *?* ont la même fonction que dans les autres interfaces. La commande *COMP-CIBLE* permet de comparer l'horaire opérationnel avec l'horaire cible du département Planification. Cette comparaison est utile pour voir dans quelle mesure l'horaire opérationnel généré des routes-horaires correspond aux recommandations du département Planification. Elle est réalisée sous la forme d'une union des deux horaires, où les services sont affichés selon une notation particulière : les services communs aux deux horaires sont affichés en vert, les services présents seulement dans l'horaire opérationnel en rouge et les services présents seulement dans l'horaire cible en jaune.

Pour retourner à l'interface 1, on utilise la commande *EFFACER* de la zone de consultation.

La zone de travail de cet interface 2 est un modèle réduit de l'interface 1. Quand cette zone est active, elle s'utilise de façon identique à l'interface 1. De plus, lorsque des routes-horaires sont modifiées dans cette zone, la zone de consultation, présentant l'horaire opérationnel généré, est mise à jour automatiquement.

3. BANQUE DE DONNÉES

Au cours de ce chapitre, nous décrivons la structure de banque de données développée pour conserver et traiter les informations de type "routing-scheduling", c'est-à-dire les horaires de service, les horaires publiés et les routes-horaires. Dans un premier temps nous présentons l'organisation générale de la banque de données. Puis nous détaillons chaque composante en apportant les justifications appropriées.

3.1 Design de la banque de données

Étant donné la petite taille du réseau physique (sept villes), la banque de données peut être placée intégralement en mémoire vive lors de l'exécution. Même lorsque nous ajouterons à la banque de données actuelle les informations relatives aux décisions des autres départements (Marketing, Achat et location, et Finances), elle restera de taille suffisamment modeste pour être chargée entièrement en mémoire[1]. Cette banque de données doit cependant être conservée sur disque entre chaque séance de simulation; ce sujet est traité au chapitre 5 de ce rapport.

Comme vu aux chapitres précédents, les données de type "routing-scheduling" se retrouvent à deux niveaux du simulateur. D'une part le département Planification établit les recommandations aux autres départements sous forme d'un horaire cible; celui-ci comprend, rappelons-le, les horaires de service cibles pour chaque paire de villes et les horaires publiés cibles pour chaque marché. Et d'autre part le département Gestion des opérations construit un "routing" (ensemble des routes-horaires) de ses véhicules duquel est généré un horaire opérationnel c'est-à-dire les

---

[1] Toutefois, certaines informations ne seront nullement nécessaires en mémoire vive et pourront être conservées sur disque pendant l'exécution. C'est le cas des données financières qui ne nécessitent aucun traitement à ce niveau du simulateur.

horaires de service opérationnels et les horaires publiés operationnels de la compagnie. Ainsi, la banque de donnnées doit contenir trois types d'informations, soient des horaires de service, des horaires publiés et des routes-horaires. Cependant elle doit aussi permettre deux sous-ensembles d'informations de même type mais regroupant des données différentes. Il s'agit de l'horaire cible et de l'horaire opérationnel qui sont tous deux composés d'horaires de service et d'horaires publiés, mais dont les contenus sont distincts.

La structure de banque de données actuellement implantée comporte les trois types d'informations requises, mais ne soutient pas deux horaires distincts. Pour les fins du stage, nous avons considéré qu'un seul horaire était suffisant pour démontrer l'efficacité de la structure en regard des différents objectifs poursuivis (cf. section 3.2). En conséquence, la banque de données actuelle comprend les routes-horaires des véhicules, les horaires de service opérationnels et les horaires publiés opérationnels. Toutefois, lors de la conception de la banque de données, nous avons tenu compte de la future intégration de l'horaire cible de manière à assurer que celle-ci se réalise sans trop de conséquences.

La section suivante présente de façon détaillée la structure de la banque de données actuelle. Il faut noter que dans cette section, nous omettons le qualificatif opérationnel des horaires pour ne parler que du type d'information en général. Les modifications envisagées pour permettre d'intégrer l'horaire cible à la banque de données de même que les implications de cette intégration sont discutées à la section 3.3.

3.2 Structure de la banque de données

3.2.1 Structure générale

Rappelons d'abord la définition des trois types d'informations considérés.

Une route-horaire est une suite chronologique de services qu'accomplit un véhicule dans une journée. Un horaire de service est une liste de services en ordre chronologique de départ, pour une paire de villes donnée. Finalement, un horaire publié est une liste de services et de services indirects en ordre chronologique de départ, pour un marché donné.

Les trois types d'informations ne manipulent donc en réalité que deux sortes de données, soient les services et les services indirects. Les différents types d'informations sont obtenus grâce à divers chemins d'accès sur ces données. Plus particulièrement, les horaires de service nécessitent d'accéder aux services en utilisant comme clé d'accès la paire O-D (i.e. la ville de départ et la ville d'arrivée du service). Les horaires publiés s'obtiennent, eux, en deux étapes, premièrement en accédant aux services en utilisant la paire O-D comme clés d'accès, puis en accédant aux services indirects avec comme clé d'accès l'itinéraire choisi pour desservir le marché. Pour les routes-horaires, l'accès doit être fait sur les services en utilisant le numéro du véhicule réalisant le service comme clé. De plus, d'autres chemins d'accès seront nécessaires pour la gestion interne de même que pour l'algorithme de génération des connexions présenté au chapitre 4 de ce rapport.

En plus de devoir fournir ces différents types d'informations, la banque de données développée doit aussi satisfaire les objectifs d'efficacité suivants :

1- permettre la consultation rapide

   a) des horaires de service
   b) des horaires publiés
   c) des routes-horaires

2- permettre la modification rapide, par l'usager

   a) des routes-horaires
   b) des horaires de services

3- permettre une mise à jour instantanée des horaires de service suite à une modification des routes-horaires.

4- permettre une mise à jour instantanée des horaires publiés suite à une modification ou une mise à jour des horaires de service.

La structure de données choisie pour l'organisation de la banque de données est une structure de multiliste. Cette structure se définit comme une collection de "cellules" dont certaines possèdent plus d'un pointeur et peuvent donc appartenir à plus d'une liste simultanément. Pour notre application, deux types de cellules sont utilisées, soient les enregistrements de services et les enregistrements de services indirects. Chaque service ou service indirect fait donc partie de plusieurs listes représentant les différentes informations de la banque de données, par exemple une liste pour la route-horaire à laquelle appartient le service, une liste pour l'horaire de service contenant ce service, etc. Ainsi, chacun des chemins d'accès décrits précédemment pour l'obtention des différents types d'informations (ou pour la gestion interne) est implanté par une liste reliant ensemble tous les enregistrements concernés ayant la même valeur pour cette clé d'accès. Par exemple, pour l'accès utilisant le numéro de véhicule comme clé, nous relions dans une même liste tous les enregistrements de services possédant le même numéro de véhicule; les listes ainsi formées correspondent aux routes-horaires des véhicules.

Cette structure de multiliste est efficace pour plusieurs raisons. Premièrement, au niveau de l'absence de redondance des données, en effet, on ne retrouve qu'une seule copie des services et services indirects malgré que ceux-ci soient utilisés pour maintes informations. Cela a l'avantage d'éviter les problèmes d'intégrité de la banque de données lors de modifications ainsi que d'économiser l'espace mémoire.

La structure de multiliste permet aussi d'accéder directement, à partir d'un enregistrement, à toutes les informations auxquelles celui-ci appartient, et ainsi d'accélérer le processus de mise à jour de la banque de données lors de modifications. De plus, chacune des listes de cette structure est implantée par un double chaînage qui accélère d'autant plus la mise à jour puisqu'on évite d'avoir à retraverser une liste pour accéder à l'enregistrement précédent. Ce double chaînage facilite aussi la consultation bidirectionnelle des diverses informations par l'usager. Cette consultation d'informations s'effectue rapidement puisqu'il ne s'agit que de parcourir la liste correspondante.

Finalement, la structure de liste, comme telle, est toute à fait appropriée pour conserver les différentes informations puisque celles-ci ne sont pas de tailles fixes. En effet, la taille des horaires de service, des horaires publiés ou des routes-horaires ne peut être déterminée a priori, car elle varie au fur et à mesure des ajouts et enlèvements effectués par l'usager. La liste est une structure flexible qui peut croître et décroître pendant l'exécution d'un programme et qui de plus est simple à mettre à jour lorsque des enregistrements sont ajoutés ou retirés.

La suite de cette section présente séparément chaque composante, c'est-à-dire chaque type de listes présentes dans la structure globale. Les justifications de la structure de liste et de multiliste ayant déjà été énoncées, les justifications supplémentaires apportées pour les composantes ne porteront que sur la spécificité de celles-ci.

3.2.2 Données

Avant de présenter les structures de chaque composante d'informations de la banque de données, voyons d'abord la définition de base des données manipulées par ces informations.

Chaque service correspond à un enregistrement comportant les éléments suivants :

- ville de départ;
- ville d'arrivée;
- heure de départ;
- heure d'arrivée;
- numéro du véhicule effectuant le service;
- numéro du service.

Chaque service indirect correspond à un enregistrement comportant les éléments suivants :

- ville de départ;
- ville d'arrivée;
- heure de départ;
- heure d'arrivée;
- numéro du service.

3.2.3 Composantes de la structure

3.2.3.1 Horaire de service

Un horaire de service est la suite des services, en ordre chronologique de départ, desservant une paire de villes donnée. Nous chaînons donc dans une même liste tous les services possédant la même paire origine-destination (O-D). Cette liste doublement chaînée pour permettre la consultation bidirectionnelle et les mises à jour aisées, est de plus ordonnée selon l'heure de départ des services. Nous utilisons une telle liste pour chaque paire de villes du réseau.

La recherche d'un élément dans une structure de liste s'effectue séquentiellement. Elle requiert donc un temps linéaire (O(n) où n représente le nombre d'enregistrements dans la liste). Ce rendement est suffisant, car les seules recherches (ou positionnements) requises dans les

listes sont pour le défilement interactif des horaires de service, où l'on déplace d'un pas prédéfini. Le système n'effectue donc aucunes recherches ponctuelles de service.

Pour accéder aux listes des horaires de service nous utilisons une matrice origine-destination (O-D) où chaque case correspond à une paire de villes. Chacune des cases contient un pointeur sur le premier service de la liste de l'horaire de service pour la paire de villes correspondante, ainsi que la distance entre ces deux villes.

La figure 3.1 présente un exemple d'horaire de service pour la paire de villes B-G.

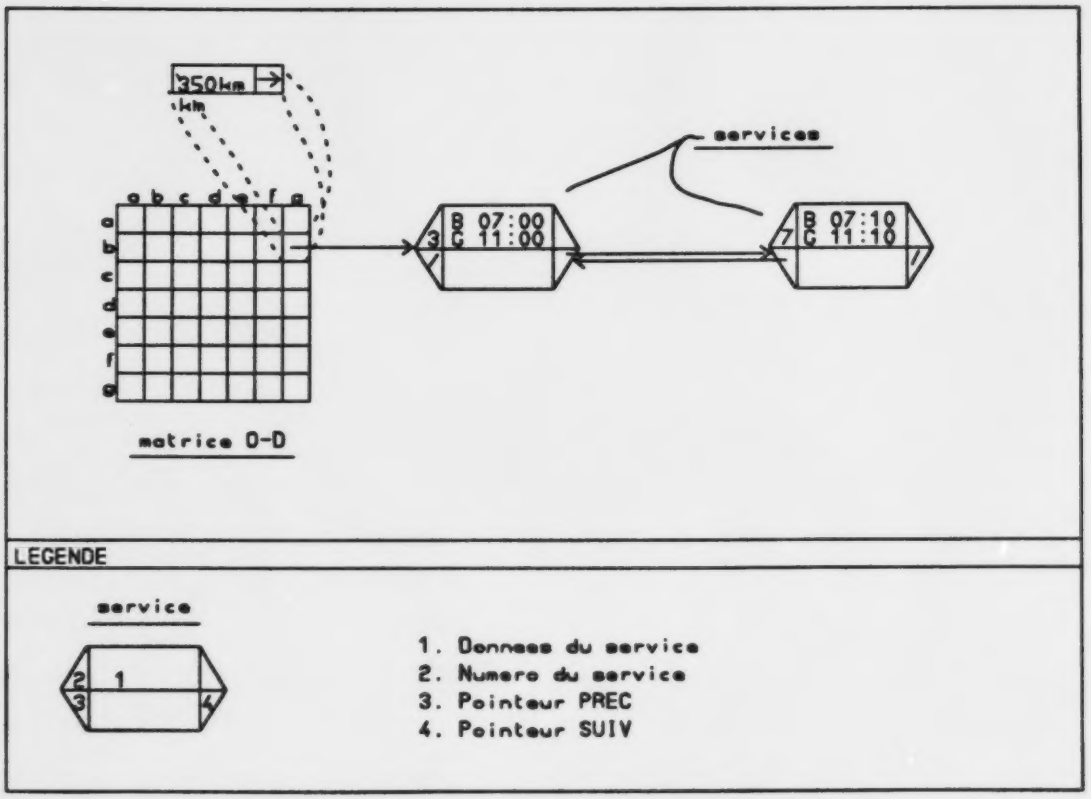


Figure 3.1 : Structure des horaires de service

Chaque service fait donc partie d'une liste doublement chaînée correspondant à l'horaire de service auquel il appartient. Pour implanter ces listes, il faut ajouter des pointeurs précédant et suivant (*PREC* et *SUIV*) à l'enregistrement du service.

3.2.3.2 Route-horaire

Une route-horaire est une suite de services qu'accomplit un véhicule dans une journée. Pour représenter la route-horaire d'un véhicule, nous joignons dans une liste doublement chaînée tous les services effectués par ce véhicule. Chaque liste est ordonnée selon l'heure de départ des services effectués. Ces listes étant en général très courtes, la recherche en temps linéaire d'un service est suffisante.

L'accès aux routes-horaires est réalisé à l'aide d'un tableau de véhicules, où chaque case contient un pointeur sur le premier service effectué par le véhicule correspondant à cette case.

La représentation schématique des routes-horaires est présentée à la figure 3.2. Les pointeurs suivants (*RSUIV*) et précédant (*RPREC*) ont été ajoutés à l'enregistrement du service pour permettre le double chaînage requis pour cette liste.

Il est à remarquer que les horaires de service et les routes-horaires pointent, grâce à la structure de multiliste, vers les mêmes données, à savoir les enregistrements de service. Aussi la mise à jour de l'horaire de service lors d'une modification de route-horaire est-t-elle facilitée. Considérons, par exemple, l'élimination d'un service d'une route-horaire. Après avoir enlevé le service de la liste correspondant à sa route-horaire, il s'agit d'enlever ce même service de la liste correspondant à son horaire de service. Cette deuxième élimination s'effectue rapidement puisqu'étant déjà positionné sur l'enregistrement du service, les services précédent et suivant de cette deuxième liste sont directement accessibles par les pointeurs *PREC* et *SUIV*.

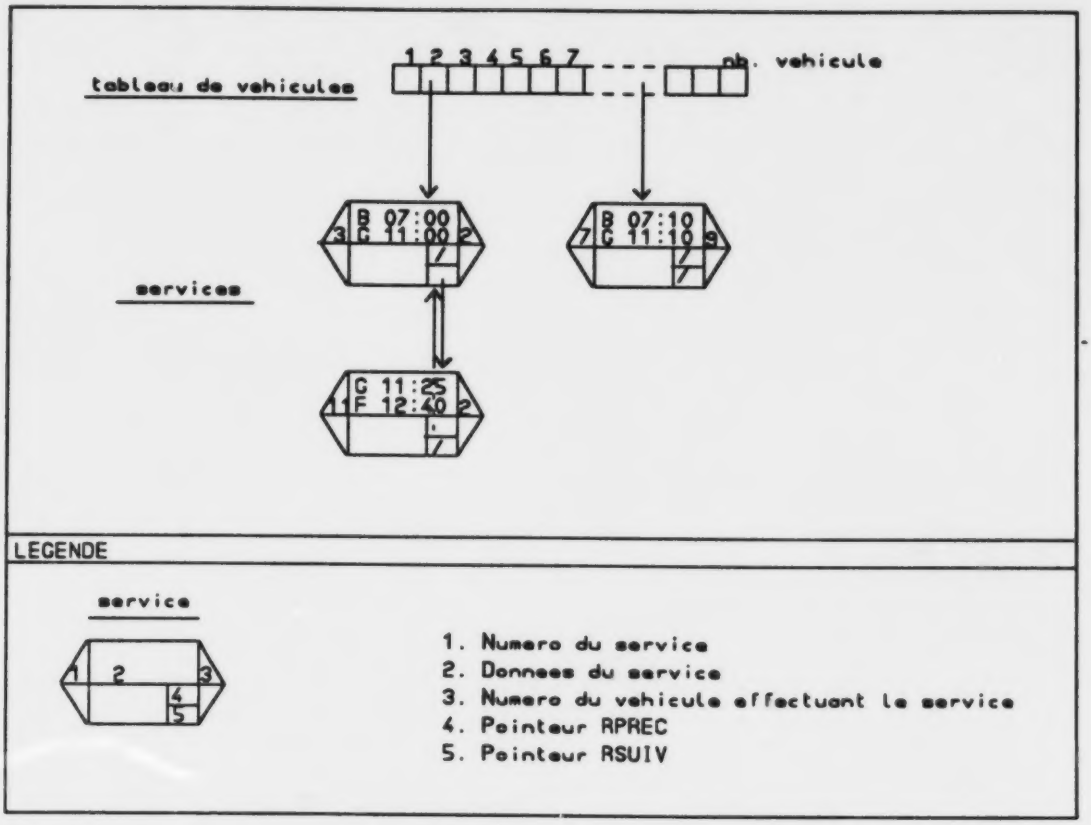


Figure 3.2 : Structure des routes-horaires

3.2.3.3 Liste d'activités

Le chapitre 4 présente l'algorithme de génération des connexions servant à construire les services indirects des horaires publiés. Cet algorithme pour être efficace nécessite un chemin d'accès différent aux services. En effet, pour former des services indirects, il faut connecter les services arrivant à une ville avec ceux en partant un peu plus tard dans la journée. Pour ce faire, nous avons besoin d'accéder au déroulement dans le temps des arrivées et départs en une ville, c'est-à-dire de pouvoir obtenir la suite ordonnée des services arrivant et quittant la ville tout au long de la journée.

Ce nouvel accès sur les services est réalisé par des listes d'activités. La liste d'activités d'une ville x contient tous les services arrivant en x ou quittant x, ordonnés selon l'heure de l'activité (arrivée ou départ) en x. Le double chaînage est nécessaire pour implanter ces listes, car l'algorithme de génération des connexions les parcourt dans les deux sens. De plus, comme déjà mentionné, le double chaînage fournit une efficacité accrue lors des ajouts et enlèvements de services.

Ces listes d'activités peuvent atteindre des tailles assez élevées. La recherche séquentielle requise pour atteindre un service particulier peut alors sembler moins efficace. Cependant, comme nous le verrons au chapitre 5, l'accès à un service d'une liste d'activités ne se réalise pas par une recherche séquentielle habituelle, i.e. débutant à la tête de la liste. Nous nous servons plutôt des services des autres listes accessibles par la structure de multiliste, pour obtenir un point d'entrée dans la liste d'activités, qui soit situé relativement près du service recherché (pour plus de détails, voir section 5.1.2).

Chaque service fait partie de deux listes d'activités, celle de sa ville de départ et celle de sa ville d'arrivée. Une liste d'activités étant doublement chaînée, il faut rajouter quatre pointeurs dans l'enregistrement du service, soit un pointeur précédent et un pointeur suivant pour chaque liste (*IPREC, ISUIV, JPREC, JSUIV*).

Finalement, l'accès aux listes est réalisé par un tableau de villes. Chaque case contient un pointeur sur le premier service de la liste d'activités de la ville correspondant à cette case.

La figure 3.3 fournit la représentation schématique de listes d'activités.

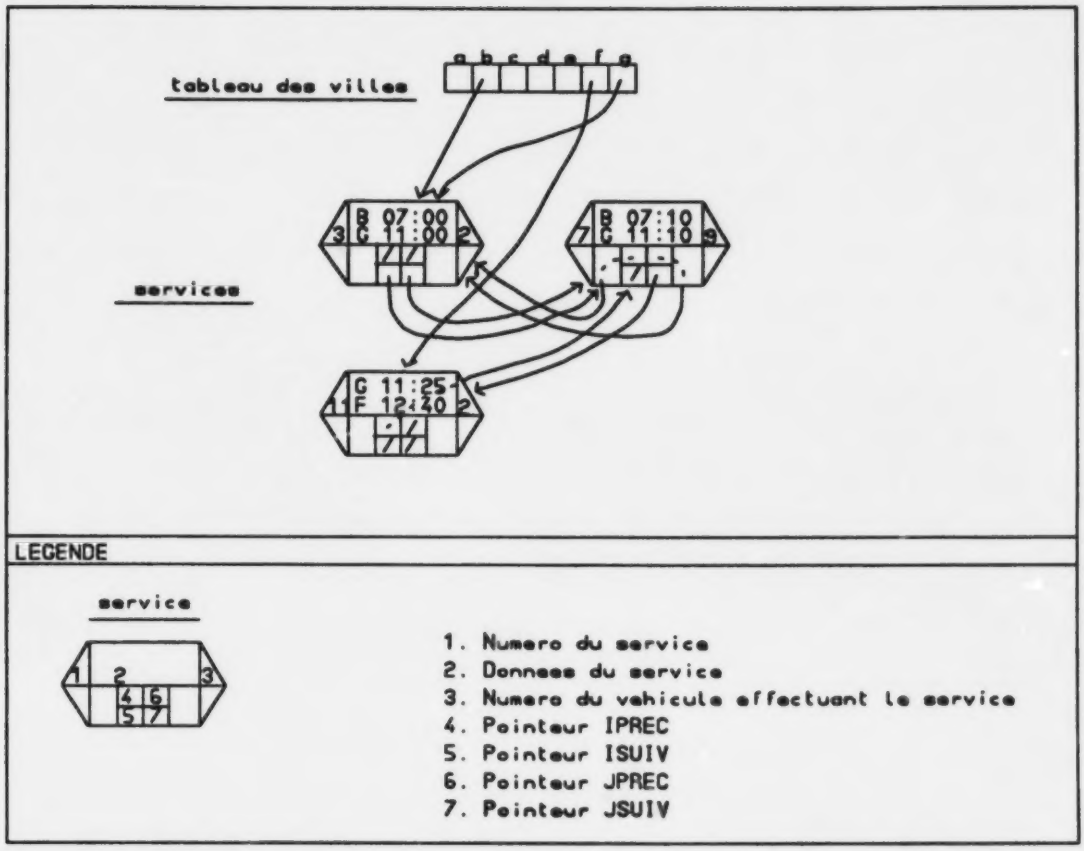


Figure 3.3 : Structure des listes d'activités

Pour parcourir la liste d'activité de la villes G, il s'agit d'abord d'accéder à la septième case du tableau de villes. Celle-ci indique le premier service de la liste, soit le service #3. Puisque ce service correspond à une activité d'arrivée en G, il faut suivre le pointeur suivant de la ville d'arrivée, soit *JSUIV*. Celui-ci nous mène au service #7, qui correspond lui aussi à une activité d'arrivée en G. Donc on suit le pointeur *JSUIV*, qui nous mène au service #11, celui-ci correspond cette fois à une activité de départ et nécessite donc de prendre le pointeur *ISUIV* pour continuer. Ce dernier étant nul la liste d'activités de la ville G est terminée.

3.2.3.4 Horaire publié

L'information horaire publié est formée par la suite des services et services indirects desservant un marché. Plus précisément, l'horaire officiel d'un marché comporte d'abord l'horaire de service de la paire de villes correspondant à ce marché, puis la liste ordonnée des services indirects pour chaque itinéraire desservant ce marché. La structure de l'horaire de service ayant déjà été présentée (cf. section 3.2.3.1), cette section portera sur la structure utilisée pour implanter les listes de services indirects.

Nous chaînons dans une même liste tous les services indirects empruntant le même itinéraire pour desservir un marché donné. Nous réalisons cela pour chaque itinéraire et pour chaque marché. De plus, chaque liste est doublement chaînée et ordonnée selon l'heure de départ des services indirects.

Pour réaliser l'accès à ces listes de services indirects, nous utilisons une matrice O-D de pointeurs sur des "index de transferts". Chaque index définit l'ensemble des villes de transferts utilisées par les différents services indirects pour desservir le marché en question (celui défini par la case de la matrice O-D). Chaque élément de cet ensemble est appelé indice; celui-ci en plus de spécifier la(les) ville(s) de transfert possède un pointeur sur le premier service indirect de la liste empruntant l'itinéraire spécifiée par la case de la matrice (origine-destination) et par l'indice (villes de transferts). Si nous utilisons une structure de tableau pour présenter un index de transfert, cela nous oblige à allouer l'espace nécessaire pour tous les transferts possibles quand seulement trois ou quatre au plus seront réellement utilisés. Pour cette raison, nous avons opté, ici aussi, pour la structure de liste. Elle est doublement chaînée pour permettre le parcours bidirectionnel lors de la consultation interactive de l'horaire.

En résumé, chaque indice de l'index comporte les données suivantes :

- une ou deux villes de transfert;
- un pointeur sur le premier service indirect de la liste (*FIRST*);
- un pointeur précédent et un pointeur suivant dans l'index (*PREC* et *SUIV*).

La figure 3.4 fournit la représentation schématique des listes de services indirects du marché C-G.

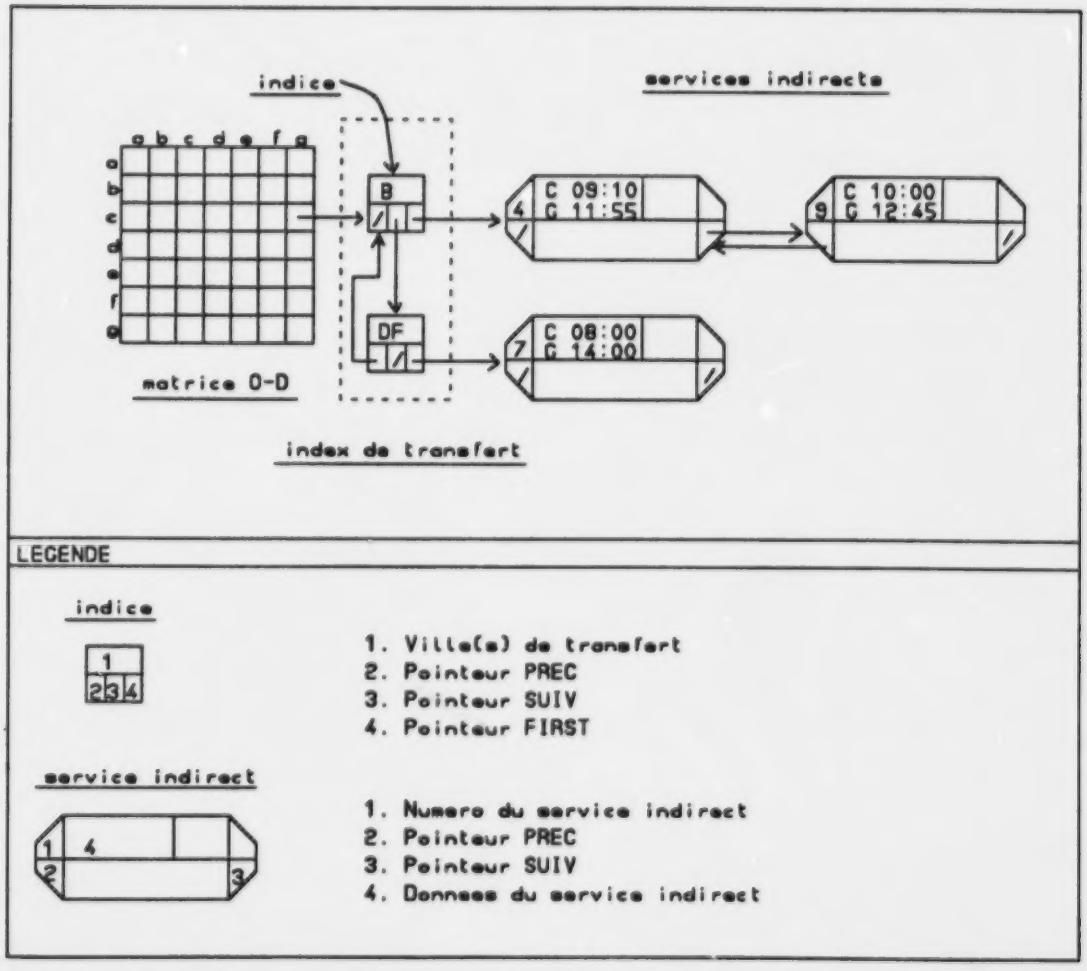


Figure 3.4 : Structure (partielle) des horaires publiés

3.2.3.5 Gestion interne

Chaque service indirect possède un ou deux transferts et est donc formé de deux ou trois services. L'enregistrement du service indirect ne comporte toutefois aucune donnée sur ces derniers. Cela évite une répétition d'informations qui gaspille l'espace mémoire et risque d'entraîner des problèmes d'intégrité de la banque de données. Cependant, il faut pouvoir accéder facilement à ces services, que ce soit pour obtenir l'heure d'un transfert ou pour savoir quel(s) véhicule(s) effectue(nt) le service indirect.

Pour résoudre ce problème, nous avons inclus dans l'enregistrement du service indirect trois pointeurs, un pour chaque service le composant *(SERV1, SERV2 et SERV3)*. Si un service indirect n'est formé que de deux services, le pointeur *SERV3* sera évidemment nul.

Avec ces pointeurs, nous pouvons, à partir de tout service indirect, atteindre directement les services le composant. Reste à résoudre le problème inverse : comment atteindre, à partir de tout service, l'ensemble des services indirects l'utilisant? Un accès rapide à cet ensemble est essentiel pour assurer une mise à jour instantanée de la banque de données lors de l'enlèvement d'un service.

Pour réaliser cet accès, nous devons relier dans un même ensemble tous les services indirects utilisant un même service, et ce pour tous les services de la banque de données. Un service indirect peut être formé d'au plus trois services, il fera donc partie d'au plus trois ensembles, i.e. un pour chaque service le composant.

Pour des raisons algorithmiques (cf. la génération des connexions, chapitre 4), il faut pouvoir distinguer la position de chaque service à l'intérieur des différents services indirects à un transfert qui l'incluent. En termes plus précis, il s'agit d'accéder séparément aux trois ensembles suivants :

GAUCHE : ensemble des services indirects à un transfert où le service forme le premier segment, i.e. la partie gauche;

DROITE : ensemble des services indirects à un transfert où le service forme le deuxième segment, i.e. la partie droite;

DEUX : ensemble des services indirects à deux transferts utilisant le service, peu importe sa position.

Pour accéder à ces trois ensembles, nous ajoutons trois pointeurs à l'enregistrement du service, soient *GAUCHE*, *DROITE* et *DEUX*.

La structure de liste est utilisée pour représenter les ensembles de services indirects définis ci-dessus. Cette structure est appropriée puisque la taille de ces ensembles varie pendant l'exécution. De plus, nous utilisons le chaînage double car il permet lors de l'enlèvement d'un service indirect, de l'éliminer rapidement de chacun des ensembles auxquels il appartient, sans avoir à retraverser les listes correspondantes. Notons cependant qu'aucun ordre n'est requis dans ces listes exclusivement utilitaires, où aucune recherche n'est effectuée. L'insertion de nouveaux services indirects dans ces ensembles s'effectue donc en tête de liste.

Pour chaque service indirect nous avons donc défini six pointeurs *PREC1*, *SUIV1*, *PREC2*, *SUIV2*, *PREC3* et *SUIV3*, soient deux pour chaque ensemble auquel il appartient.

La figure 3.5 présente les liens existants entre les services et services indirects. Si l'on suit par exemple le pointeur *GAUCHE* du service #6, on atteindra l'ensemble des services indirects où ce service forme le premier segment, i.e. la partie gauche. On retrouve ainsi premièrement le service indirect #20, à partir de celui-ci on rejoint le #11 grâce au pointeur *SUIV1*. Le pointeur *SUIV1* du service indirect #11 étant nul, l'ensemble est terminé. À noter que chaque pointeur *SEHV1* des services indirects de cet ensemble réfère bien le service #6.

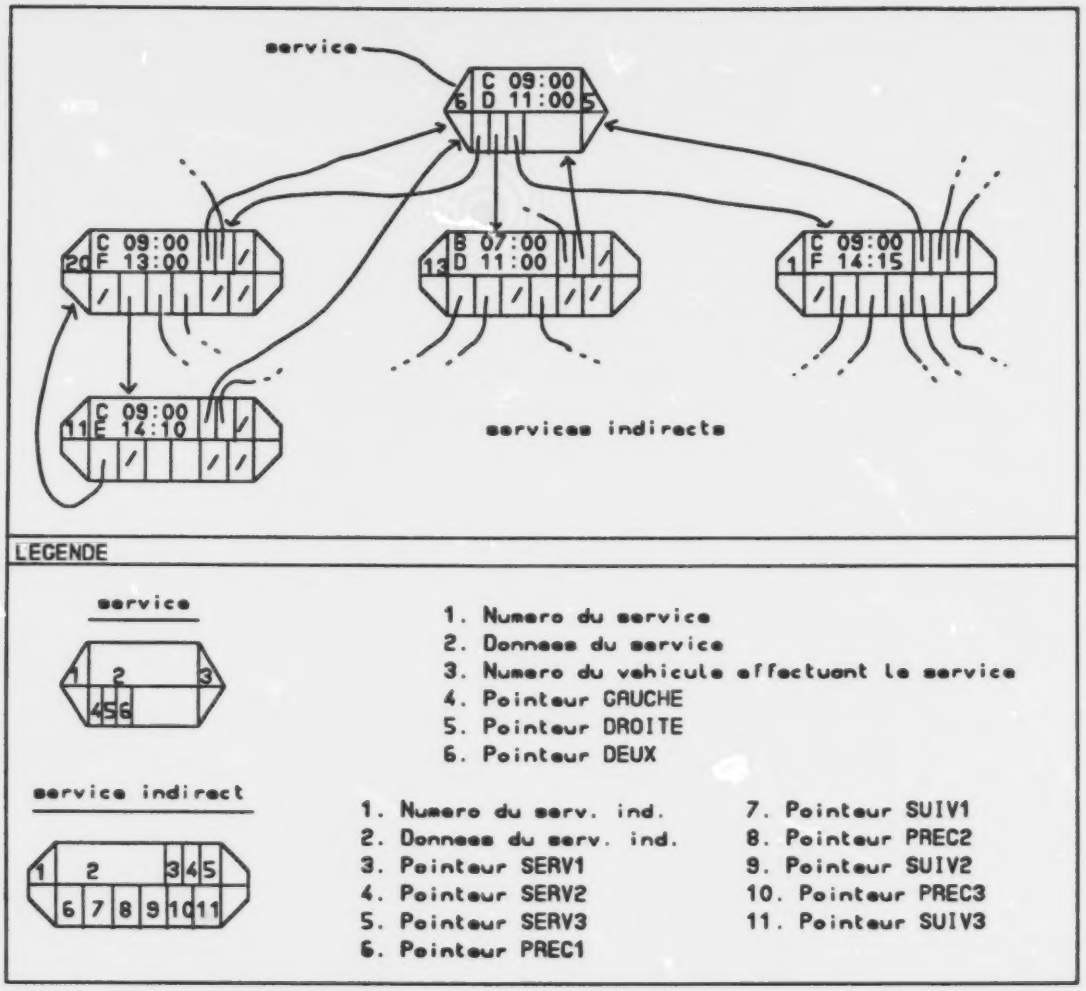


Figure 3.5 : Liens existants entre services et services indirects

3.2.4 Intégration

L'appendice A présente la déclaration de la structure de la banque de données. Il s'agit de la définition complète en langage C des services et services indirects, avec tous les pointeurs, ainsi que de la déclaration de toutes les structures d'accès sur ces services.

La figure 3.6 présente le schéma global de la structure de multi-liste de la banque de données. On y retrouve toutes les composantes décrites dans les sections précédentes. Peu de données ont été représentées pour des raisons évidentes de lisibilité.

3.3 Modifications envisagées

Comme mentionné au début de ce chapitre, la banque de données finale devra contenir les données de l'horaire cible du département Planification. Le but de cette section est de présenter les modifications et ajouts prévus à la structure de la banque de données pour intégrer cet horaire cible, ainsi que leurs implications.

L'horaire cible et l'horaire opérationnel étant des informations de même type, mais regroupant des données différentes, il est évident que la structure utilisée pour une soit la même que pour l'autre. Ainsi les enregistrements des services et services indirects cibles (i.e. provenant de l'horaire cible) sont identiques à ceux présentés à la section précédente (cf. section 3.2.2). Toutefois, le champ numéro de véhicule de l'enregistrement de service est inutile pour l'horaire cible. En effet, le département Planification n'assigne aucun véhicule aux services qu'il construit. Ainsi ce champ sera toujours nul.

Nous considérons cette solution préférable à celle de définir un nouvel enregistrement de service ne comportant pas ce champ, car la perte d'espace encourue par la première solution est minime comparée aux complications nécessitées par l'obligation de considérer deux types d'enregistrement de service lors de la programmation.

Au niveau des accès à l'horaire cible, les modifications prévues sont les suivantes. On ajoutera aux éléments de la matrice O-D permettant l'accès aux horaires de service, un pointeur pour accéder aux horaires de service cibles. Pour les listes d'activités, on ajoutera ici aussi un

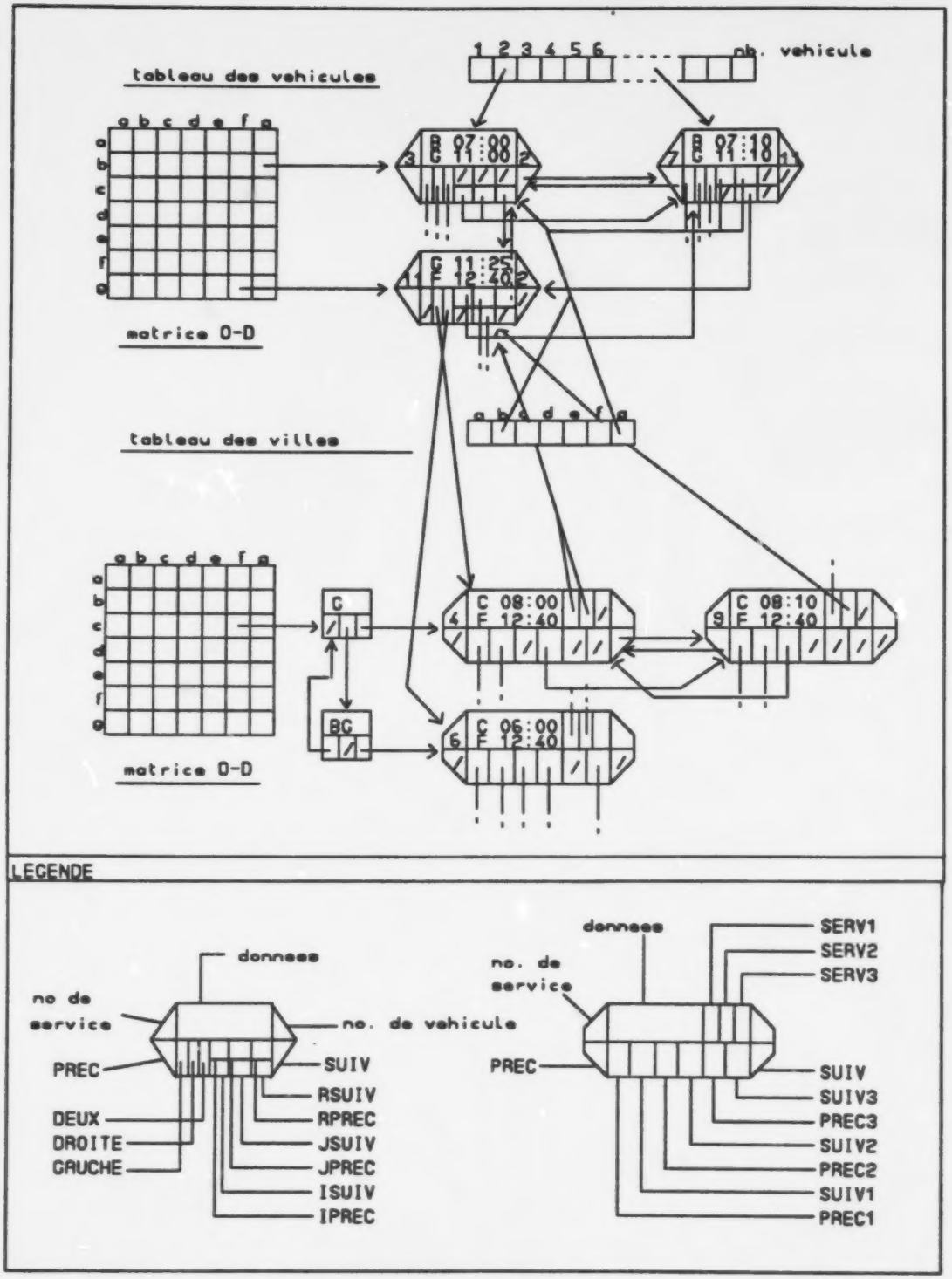


Figure 3.6 : Structure de multiliste de la banque de données

pointeur à chaque case du tableau de villes pour permettre de rejoindre les listes d'activités des services cibles. Pour les horaires publiés, ou plutôt les listes de services indirects, la matrice O-D accédant aux index de transferts permettra aussi l'accès à des index de transferts menant aux listes de services indirects cibles. Évidemment, il y aura entre les services cibles et les services indirects cibles les mêmes liens de gestion interne que ceux présentés à la section 3.2.3.5.

En conséquence, la taille de la banque de données aura pratiquement doublée lorsque ces modifications auront été effectuées. Cependant, le réseau étant de petite taille, aucun problème d'espace n'est prévu.

Finalement, les données non opérationnelles en provenance des autres départements devront aussi être intégrées soit à la banque de données, soit dans des fichiers à part, dépendant de l'espace disponible. Puisque ces données ne subissent aucun traitement à ce niveau du simulateur l'une ou l'autre méthode est aussi valide.

3.4 Expérimentation

La structure de la banque de données s'est avérée efficace lorsque nous l'avons implantée et testée. Elle répond aux objectifs de rapidité exigés pour notre système. Des banques de données de taille normale (autour de 700 à 800 services et services indirects) ont été opérées sans expérimenter d'attente entre les commandes effectuées.

Un nombre appréciable de pointeurs a été nécessaire pour atteindre cette rapidité d'exécution. Toutefois, l'espace mémoire accaparé par ceux-ci n'est pas problématique étant donné la taille très modeste du réseau de villes utilisé dans le simulateur.

Évidemment cette structure de banque de données est spécialement appliquée à des réseaux de petites tailles. Elle n'est donc pas adaptée à l'exploitation réelle sur des réseaux de grandes tailles, mais est plutôt destinée au domaine de l'apprentissage qui emploie des réseaux de tailles beaucoup plus modestes.

4. GÉNÉRATION DES CONNEXIONS

Tout au long de son processus d'analyse et de prise de décision, le participant a accès à l'horaire publié de chaque marché. Cet horaire est composé des services offerts sur la paire de villes correspondante au marché, ainsi que des services indirects sur ce marché. Ces derniers sont formés par des connexions entre les services.

La création et la mise à jour des horaires publiés sont l'objet du présent chapitre. Nous discutons d'abord de la problématique du choix des connexions à réaliser pour l'obtention de services indirects *plausibles*. Nous présentons en particulier la notion de *plage horaire* développée pour résoudre ce problème. Puis, nous décrivons comment cette notion est intégrée et employée dans l'algorithme de génération des connexions. Finalement, nous donnons le détail de cet algorithme conçu pour une mise à jour instantanée des horaires publiés, suite à l'insertion d'un nouveau service dans la banque de données.

4.1 Description de la problématique

Un horaire publié sur un marché donné est constitué de l'ensemble des services desservant ce marché et d'un certain nombre de services indirects. Les services inclus dans cet horaire sont évidemment ceux de l'horaire de service sur la paire de villes correspondante au marché. Les services indirects, quant à eux, doivent être formés par la combinaison de services de façon à desservir indirectement le marché spécifié. Les services indirects considérés sont ceux provenant de la combinaison de deux ou trois services.

La génération de tous les services indirects possibles, même pour un réseau de petite taille, produirait un horaire publié énorme, qui contiendrait, de plus, maints services indirects peu intéressants pour le

voyageur. Cette génération serait également trop lente pour le système interactif recherché. Par conséquent, nous voulons développer une procédure de génération des connexions entre services qui ne produisent qu'un ensemble restreint de services indirects, et qui de plus correspondent aux services indirects "logiquement" considérés par les voyageurs.

L'analyse détaillée de la problématique qui suit, de même que l'approche de résolution présentée plus loin repose sur une hypothèse de comportement rationnel des voyageurs qui tient compte de la diversité des préférences quant aux tarifs, aux heures de départ, aux durées de voyage, aux temps d'attente dans les villes de transfert et aux heures d'arrivée. Cette hypothèse suppose que, abstraction faite des tarifs, les voyageurs considèrent plusieurs itinéraires géographiquement plausibles et que pour chacun de ces itinéraires, ils considèrent les combinaisons de services satisfaisant au mieux leurs heures de départ ou d'arrivée souhaitées. Parmi ces combinaisons, chaque voyageur choisit celle qui minimise la durée de son voyage, et donc le temps d'attente au(x) ville(s) de transfert, ou bien l'une des combinaisons "voisines" si celle-ci s'avère indisponible faute de place.

4.1.1 Choix des itinéraires

Un service indirect est constitué de deux ou trois services qui relient l'origine d'un marché à sa destination, selon un certain itinéraire. Il s'agit de déterminer quels itinéraires doivent être considérés lors de la formation des services indirects. Considérer toutes les possibilités serait long et dispendieux. Ce serait d'ailleurs inutile puisque certaines villes sont manifestement mal localisées pour servir de transfert à un marché donné; aucun voyageurs n'emprunteraient les services indirects générés sur ces itinéraires.

L'itinéraire d'un marché correspond à une suite de villes, soient l'origine, une ou deux villes de transfert et la destination. Puisque le réseau des villes est fixe, nous pouvons spécifier a priori l'ensemble des villes de transfert *plausibles* pour chaque marché. Ceci permet de simplifier la génération des services indirects, tout en nous assurant du

"réalisme" des itinéraires considérés. Les villes incluses dans cet ensemble sont celles permettant de former un itinéraire intéressant pour le voyageur, c'est-à-dire un itinéraire relativement court.

Toutefois, cette solution peut s'avérer contraignante pour des participants désireux d'expérimenter l'effet de tarifs très bas sur des itinéraires géographiquement inintéressants. Pour cette raison, le système développé fournit de façon automatique tous les itinéraires qui incluent les transferts définis comme plausibles, mais il offre aussi une procédure interactive permettant d'augmenter l'ensemble de ces villes de transfert. Cette fonction devrait toutefois rester d'utilisation marginale, puisque nous estimons avoir inclu dans cet ensemble tous les itinéraires normalement considérés par les voyageurs.

La génération des services indirects est ainsi réduite à ceux suivant des itinéraires plausibles ou désirés par le participant.

4.1.2 Choix du critère de connexion

Le problème considéré ici porte sur les connexions à établir entre l'ensemble des services arrivant à une ville x donnée - correspondant à une ville de transfert plausible pour le marché considéré - et l'ensemble des services quittant cette ville.

Considérons le schéma d'arrivées et de départs (schéma A/D) de la figure 4.1. Les connexions les plus intéressantes sont 3+6 (i.e. le service #3 avec le service #6), 4+9 et 5+10, car elles minimisent le temps d'attente à la ville de transfert. Toutefois, lorsque des problèmes de capacité surviennent, ou pour tenir compte de la volonté de certains voyageurs de demeurer plus longtemps dans la ville de transfert, d'autres connexions doivent être offertes. Il est néanmoins exclu de générer toutes les connexions possibles, surtout qu'elles peuvent s'avérer très nombreuses et que certaines sont manifestement plus intéressantes que d'autres. Par

exemple, si le service #6 est complet, la connexion 3+7 est une deuxième option intéressante pour un voyageur sur le service #3. Tandis que plus on s'éloigne de l'arrivée du service #3 pour établir une connexion, moins celle-ci sera attrayante pour le voyageur. Aussi une connexion tel que 3+10 est-elle peu vraisemblable et plutôt inutile à générer.

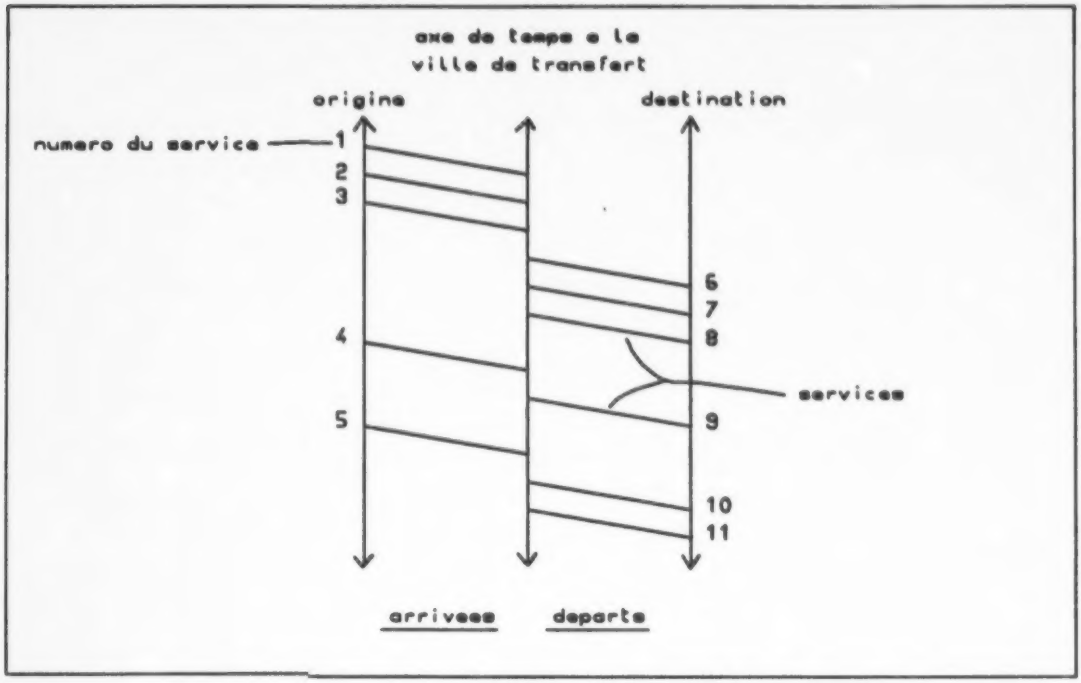


Figure 4.1 : Schéma d'arrivées et de départs (A/D)

Nous voulons définir un critère qui permette de générer les connexions attrayantes pour le voyageur. Étant donné que cette notion d'attrayance est vaste et varie beaucoup d'un voyageur à l'autre, nous ne prétendons pas pouvoir générer l'ensemble de connexions idéal, mais tout simplement de limiter la taille des horaires publiés, et donc aussi leur temps de génération, en n'incluant que les connexions les plus normales, en se basant sur le comportement du voyageur type décrit en début de chapitre. De plus, rien n'empêche un voyageur de considérer d'autres connexions que

celles générées, il n'a qu'à consulter les services annoncés dans les horaires publiés.

Un premier critère possible reposant sur le temps peut par exemple être défini comme suit : *"chaque service arrivant à une ville est connecté avec tous les services quittant cette ville dans les deux heures suivant l'arrivée"*. Un tel critère pourra donner de bons résultats autant que de mauvais, dépendant de la disposition des arrivées et départs. Si par exemple pour une arrivée donnée l'intervalle de deux heures se termine à 13:15 et qu'il y a seulement deux départs, un à 13:14 et l'autre à 13:16, ce serait une erreur de n'annoncer que la connexion avec le premier départ, les voyageurs étant tout aussi intéressés par le deuxième.

Un autre critère, tout aussi arbitraire, est celui reposant sur le nombre de services rencontrés. Un exemple serait : *"chaque service arrivant à une ville est connecté avec les trois prochains services quittant cette ville"*. Ici aussi, la qualité des connexions dépend de la disposition des arrivées et départs, la troisième connexion pouvant par exemple correspondre à un temps d'attente supérieur de plusieurs heures aux deux autres connexions.

Le critère recherché doit simuler au mieux le comportement des voyageurs. Ceux-ci veulent minimiser le temps total du voyage. Donc, sur un itinéraire donné, le temps de transport étant fixe, il s'agit de limiter le temps d'attente à la ou aux villes de transfert. Mais ne publier que la "meilleure connexion" est insuffisant, car des problèmes de congestion peuvent survenir, et d'autres options doivent alors être disponibles. Voyons donc ce que pourrait décider un voyageur type sur le schéma A/D de la figure 4.1.

> *"En principe un voyageur sur le service #3 désire emprunter le service #6. Si celui-ci est plein, il considère le #7. Si celui-ci est aussi plein, il considère le #8, mais si celui-ci s'avère également plein, le voyageur ne considérera pas la connexion 3+9, car il serait alors préférable pour lui de retarder son heure de départ en prenant le service #4 comme premier segment. Avec une connexion sur le service #9, il diminue son temps d'attente et donc le temps total de son voyage."*

En résumé, quant la "meilleure connexion" envisagée par un voyageur s'avère impossible, les choix considérés sont ceux situés après, en ordre décroissant d'intérêt plus ou s'éloigne de la connexion initialement souhaitée, cela jusqu'à ce qu'on rencontre une autre "meilleure connexion". C'est ce raisonnemnt qui ait implicitement tenu dans la notion de "plage horaire" proposée à la section suivante.

Remarque : Dans des cas extrêmes où toutes les connexions attrayantes seraient congestionnées, le voyageur n'aura pas d'autres choix que de prendre des connexions non annoncées dans l'horaire publié du marché. Ce cas sera considéré pendant la simulation.

4.2 Notion de plage horaire

Le critère de formation des connexions, développé pour la génération des services indirects des horaires publiés, subdivise les arrivées et les départs en plages horaires.

Une plage horaire d'arrivée (PA) est une succession d'arrivées en une ville qui n'est pas interrompue par aucun départ.

Une plage horaire de départ (PD) est une succession de départs en une ville qui n'est pas interrompue par aucune arrivée.

La figure 4.2 présente les différentes plages du schéma A/D de la figure 4.1.

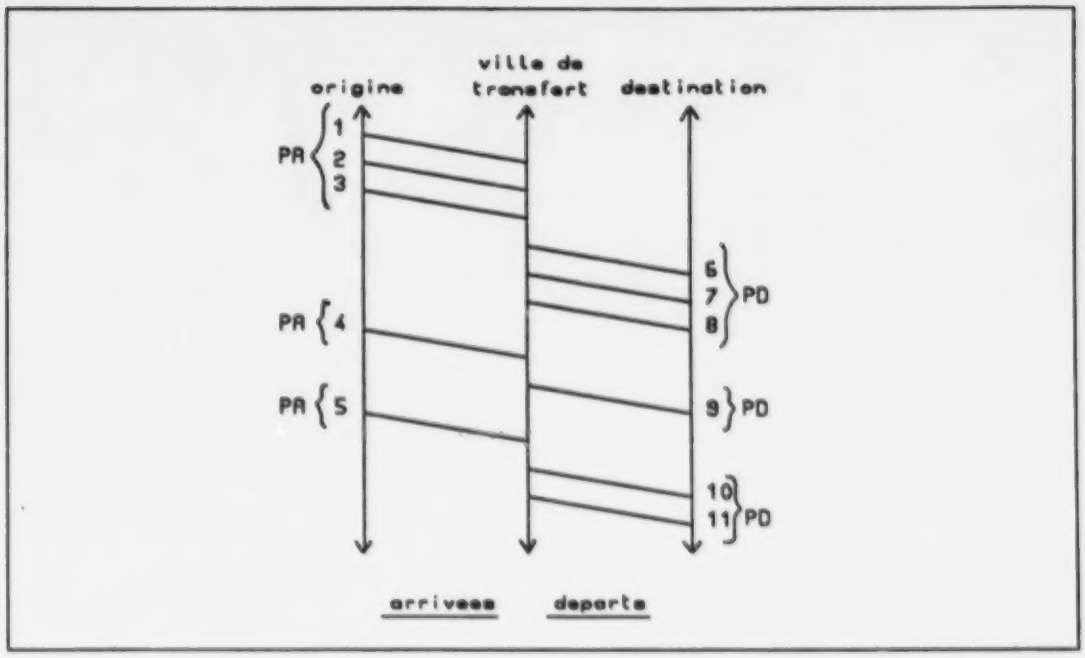


Figure 4.2 : Plages horaires

Ces plages horaires sont définies pour un marché spécifique, i.e. que tous les services des plages d'arrivées proviennent de la même ville et tous les services des plages de départ ont la même destination (cf. figure 4.2).

Le critère sur lequel repose la génération des connexions établit que tous les services d'une plage d'arrivée doivent être combinés avec tous les services de la plage de départ adjacente. Ainsi pour le schéma A/D de la figure 4.2, nous obtenons 12 connexions sur une possibilité de 30 (5 arrivées par 6 départs), soient 1+6, 1+7, 1+8, 2+6, 2+7, 2+8, 3+6, 3+7, 3+8, 4+9, 5+10 et 5+11. Ces connexions pour être valides doivent satisfaire d'autres contraintes, telles faire partie de l'intervalle de temps où une connexion est possible, ou bien suivre un itinéraire plausible. Ces autres contraintes sont traitées à la section 4.3.2.

Pour un schéma A/D plus "balancé", le critère de plage donne un ensemble de connexions valides très différent. Par exemple, pour le schéma A/D de la figure 4.3, seulement quatre connexions sont produites, soient

1+5, 2+6, 3+7 et 4+8. Peu d'options sont offertes au voyageur pour pallier à d'éventuels problèmes de congestion. Néanmoins, il reste que si les service #5 est plein, le voyageur prendra la connexion 2+6, qui est effectivement annoncée dans l'horaire publié de ce marché, sur la base de notre critère de plage. Ce critère satisfait donc bien le but de l'horaire publié qui est d'indiquer les choix de services indirects intéressants pour le voyageur, et non pas de traiter tous les cas possibles. Si, lors de la simulation des déplacements des voyageurs sur le réseau, des problèmes de congestion subsistent, des connexions additonnelles, telles 1+6, pourront alors être considérées.

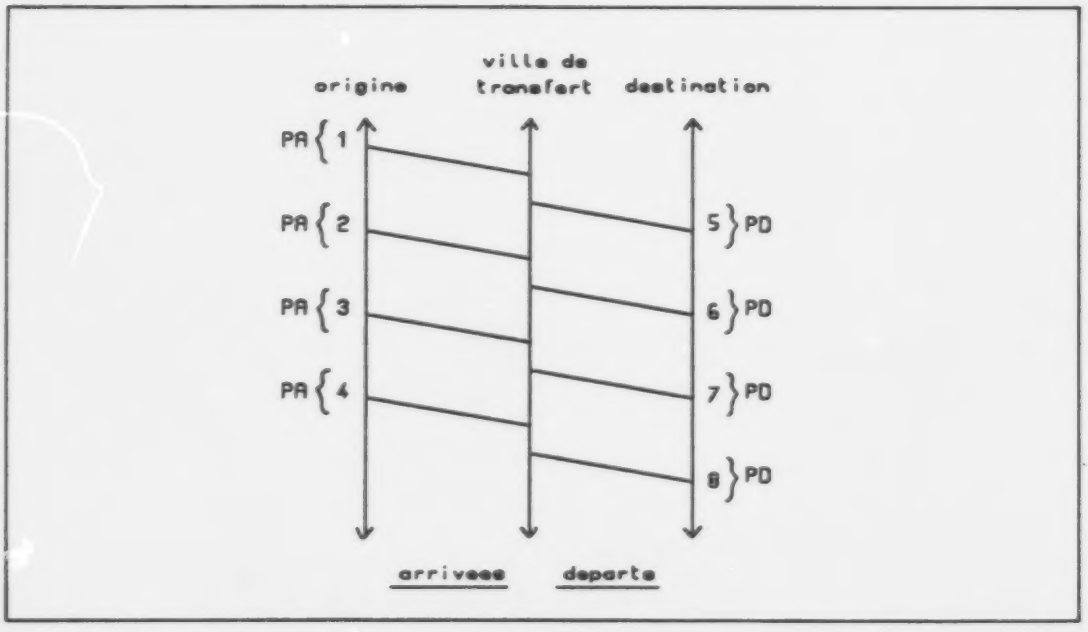


Figure 4.3 : Schéma A/D balancé

En résumé, la génération des connexions selon le critère de plage horaire réflète bien la logique de comportement des voyageurs : elle tient compte d'une part de l'objectif de minimisation du temps d'attente et d'autre part des problèmes éventuels de congestion.

Notons finalement que ce principe de connexion s'applique dans la génération des services indirects à un et deux transferts. Pour ces

derniers, chacune des deux connexions doit répondre au critère des plages horaires.

4.3 Génération des connexions

L'algorithme de génération des connexions développé pour le simulateur utilise indirectement le principe de connexion des plages horaires décrit à la section précédente; celui-ci est adapté à l'aspect dynamique et interactif du système de manière à en accroître l'efficacité au niveau de la rapidité d'exécution.

4.3.1 Perspective dynamique

Le système étant interactif le processus de génération des connexions est activé pour chaque service ajouté à la banque de données. Toutefois, une combinaison complète à chaque fois des plages d'arrvées et de départs adjacentes produirait des services indirects déjà existants. Il faut plutôt combiner uniquement le nouveau service avec ses plages adjacentes de départs et d'arrivées.

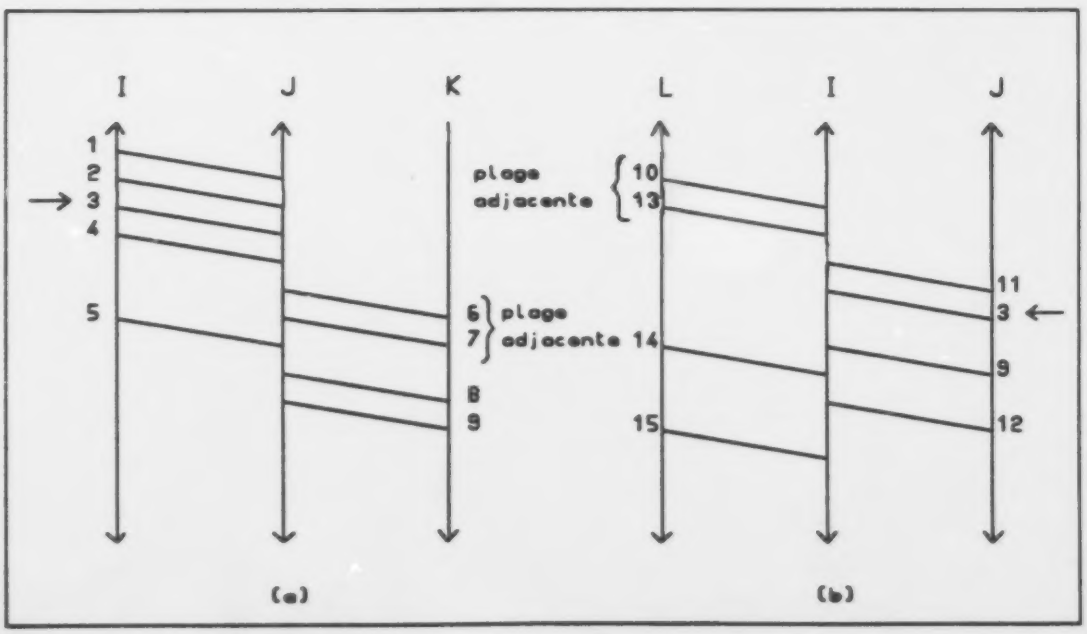


Figure 4.4 : Schémas A/D contenant un nouveau service

Supposons que l'on ajoute un service #3 au réseau. La figure 4.4(a), qui représente le schéma A/D à la ville d'arrivée de ce nouveau service, indique que sa plage adjacente est formée par les services #6 et #7. La figure 4.4(b), qui présente le schéma A/D à la ville de départ du service #3, indique que la plage adjacente contient les services #10 et #13. Bref, à la ville d'arrivée, la plage adjacente est la première plage de départ rencontrée en "descendant" dans l'axe de temps, tandis qu'à la ville de départ, la plage adjacente est le première plage d'arrivée rencontrée en "remontant" dans l'axe de temps. Dans notre exemple, l'ajout du service #3, engendre la création des services indirects (3+6) (c'est-à-dire celui formé de la connexion 3+6), (3+7), (10+3) et (13+3).

L'exemple de la figure 4.4 simplifie toutefois la situation, car il ne considère qu'une origine pour les plages d'arrivées et qu'une destination pour les plages de départ. En fait, l'algorithme doit connecter le nouveau service avec les services des plages adjacentes relatives à différentes villes origines et destinations.

Pour un service arrivant dans une ville j, il y a six plages adjacentes possibles, soient une vers chacune des six autres villes considérées dans le simulateur. De même pour un service quittant une ville i, il y a six plages adjacentes possibles, i.e. une provenant de chaque ville sauf i. Nous désignons sous le nom de **"plage de départ adjacente généralisée"** l'ensemble des plages de départ adjacentes pour les six destinations possibles. Une **"plage d'arrivée adjacente généralisée"** est définie de façon analogue. Ces notions seront précisées et utilisées dans la prochaine section lors de la présentation de l'algorithme de génération des connexions.

Le continuel changement des plages horaires suite à des ajouts ou enlèvements de services est une autre conséquence de l'aspect dynamique du système. Puisque les services indirects sont le résultat de connexions

contraintes à respecter le critère des plages horaires, nous devons nous assurer de leur validité et procéder aux modifications nécessaires.

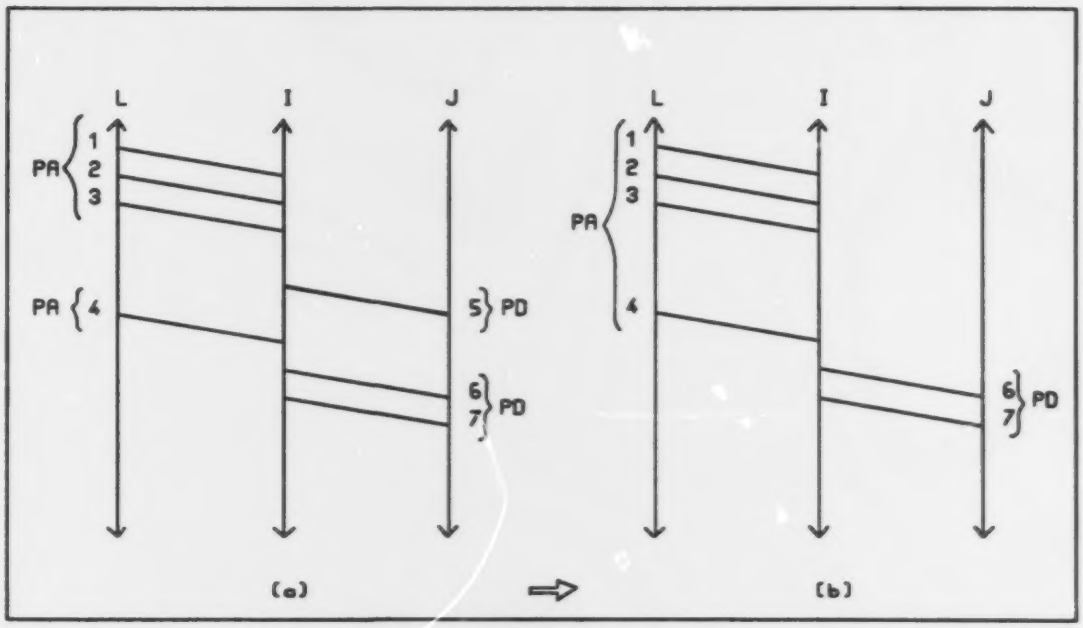


Figure 4.5 : Schémas A/D avec enlèvement de service

La figure 4.5(a) présente le schéma A/D à la ville de départ du service #5. Si ce service est enlevé de la banque de données, il faut évidemment éliminer les services indirects l'utilisant, soient (1+5), (2+5) et (3+5). Cependant, d'autres mises à jour sont également requises, dû au nouvel état des plages horaires causé par l'enlèvement du service (cf. figure 4.5(b)). Cette réorganisation entraîne la création des services indirects respectant ces nouvelles plages, soient (1+6), (1+7), (2+6), (2+7), (3+6) et (3+7). De la même façon, l'enlèvement du service #5 a pu modifier la disposition des plages horaires à la ville d'arrivée de ce service.

De même, l'insertion d'un nouveau service n'entraîne pas seulement la génération de nouveaux services indirects. Elle peut aussi modifier la

disposition des plages et provoquer l'élimination de services indirects déjà existants.

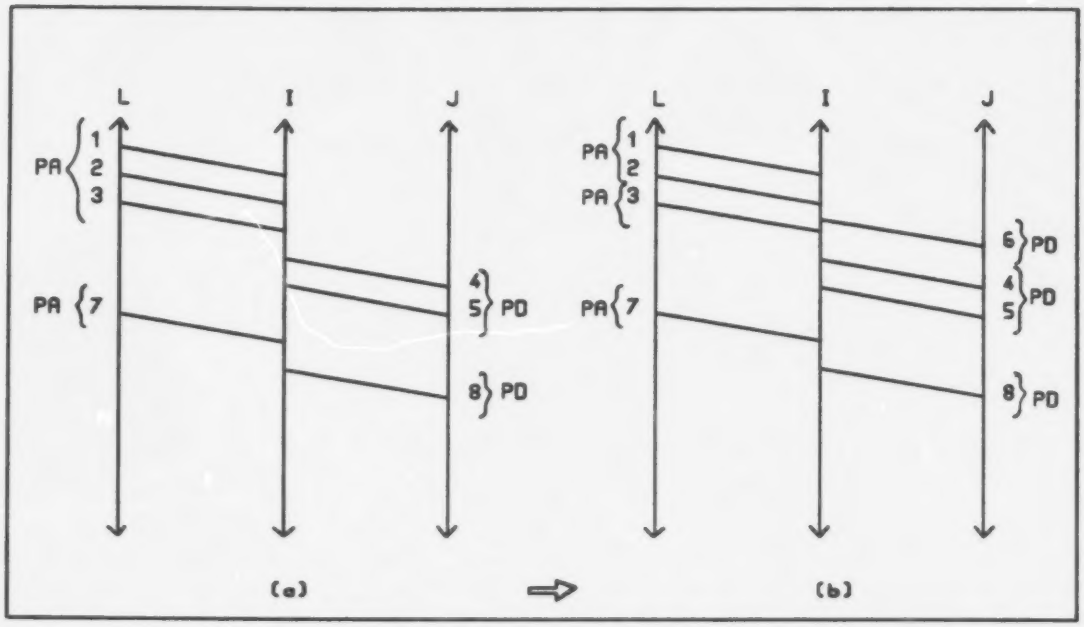


Figure 4.6 : Schémas A/D avec insertion de service

Ainsi, dans la figure 4.6, l'insertion du service #6 entraîne la génération des services indirects (1+6) et (2+6), mais cause l'élimination des services indirects (1+4), (2+4), (1+5) et (2+5). Ces derniers n'étant plus valides dans la nouvelle disposition des plages. De la même façon l'ajout du service #6 peut modifier les plages de départ à la ville d'arrivée du service.

Le chapitre 5 présentera en détail le traitement effectué lors de ces réorganisations de plages horaires. Le reste de ce chapitre présente simplement l'algorithme de génération des services indirects pour un nouveau service.

4.3.2 Algorithme

4.3.2.1 Principe général

L'algorithme doit construire tous les services indirects valides à

un ou deux transferts, utilisant un service donné. Les sections précédentes ont indiqué que cette construction doit générer les connexions à la ville de départ et à la ville d'arrivée du service ajouté, et doit de plus atteindre tous les marchés concernés.

Les listes d'activités, introduites au chapitre 3, permettent d'accéder au déroulement dans le temps des arrivées et départs en une ville. Ainsi par un simple parcours de ces listes, nous pouvons, pour tout service, atteindre d'une part la plage d'arrivée adjacente généralisée à la ville origine de ce service, et d'autre part, la plage de départ adjacente généralisée à la ville destination de ce service, ce qui permet de générer rapidement les connexions pour l'ensemble des marchés.

Chaque service appartient à deux listes d'activités, soient celle de sa ville de départ et celle de sa ville d'arrivée. Supposons l'insertion d'un nouveau service, SERV, dans ces listes. En parcourant vers le bas la liste d'activités de sa ville d'arrivée, nous rencontrons tous les services quittant cette ville à une heure subséquente[1]. Ainsi pour chaque service rencontré, il s'agit de tester la validité d'une connexion et éventuellement de la générer si elle satisfait toutes les contraintes. Le même processus est utilisé pour les connexions à la ville de départ.

La figure 4.7 schématise une liste d'activités à la ville B. Soit (CB, 7:00, 9:05) le nouveau service direct inséré dans la banque de données. La génération des nouvelles connexions à la ville d'arrivée B s'effectue en descendant dans la liste. Pour chaque service rencontré, où l'activité en B est un départ, une connexion peut être établie, mais cette dernière est sujette à plusieurs vérifications. En particulier, il faut vérifier la plausibilité de l'itinéraire du futur service indirect (cf.

---

1 Nous rencontrons aussi les services arrivant à cette ville à une heure subséquente. Mais ceux-ci ne nous intéressent pas pour l'instant.

section 4.1.1). Des limites sur les délais de connexions minimaux et maximaux doivent aussi être respectées. Et évidemment, la connexion doit répondre au critère de combinaison des plages horaires adjacentes.

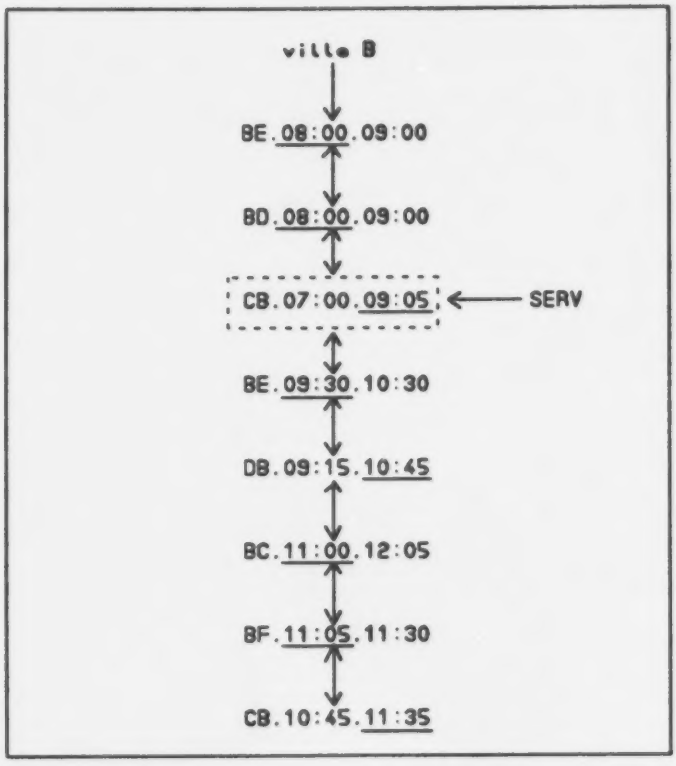


Figure 4.7 : Liste d'activité à la ville B

Supposons pour l'instant que ces conditions sont remplies, alors deux connexions sont possibles soient CB, 7:00, 9:05 + BE, 9:30, 10:30 et CB, 7:00, 9:05 + BF 11:05, 11:30. La connexion avec BC 11:00, 12:05 n'est évidemment pas considérée.

La construction des services indirects où le nouveau service SERV forme le deuxième segment s'effectue de façon similaire, mais en remontant dans la liste d'activités de la ville de départ, soit C, de manière à rencontrer tous les services arrivés en C à une heure antérieure.

4.3.2.2 Traitement des plages

Aucune plage horaire n'est conservée entre deux opérations. De plus, les seules plages considérées pendant la génération des connexions sont celles nécessaires au traitement. Nous évitons ainsi des mises à jour lourdes et fréquentes des plages qui, comme on l'a vu précédemment, se modifient constamment.

Revenons au nouveau service SERV qui doit être connecté avec la plage d'arrivée adjacente généralisée de sa ville de départ et avec la plage de départ adjacente généralisée de sa ville d'arrivée.

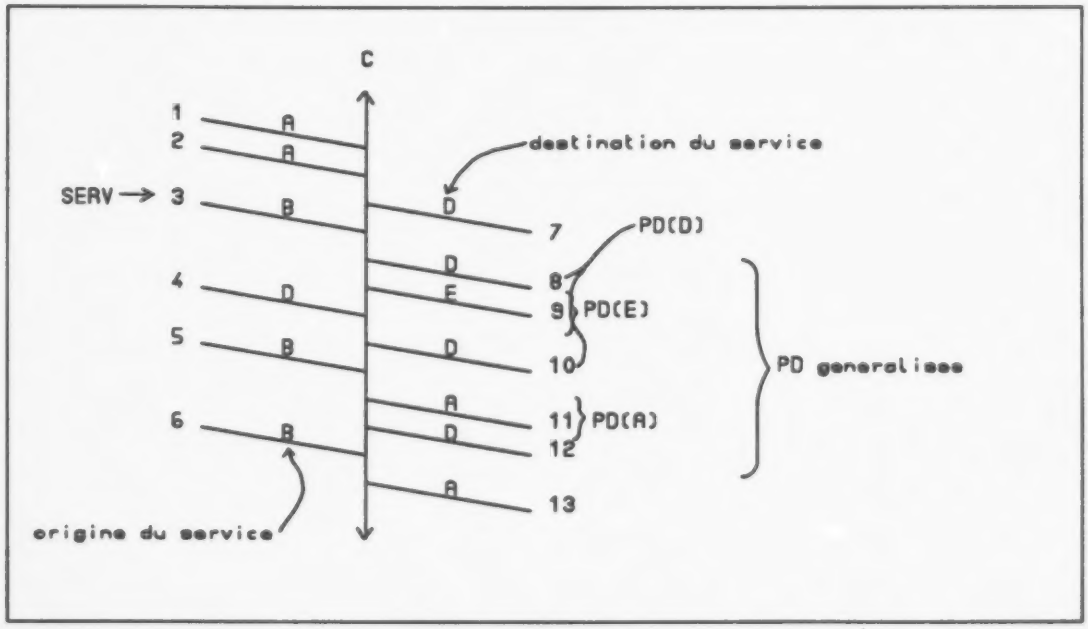


Figure 4.8 : Schéma A/D avec plage généralisée

La figure 4.8 exhibe le schéma A/D représentant la liste d'activités à la ville d'arrivée de SERV, soit C. Pour générer les connexions, nous parcourons cette liste vers le bas à partir de SERV. Supposons que toutes les contraintes de délais et d'itinéraires sont satisfaites et voyons

comment est traitée la contrainte de plage. Nous rencontrons d'abord les services #8 et #9 avec lesquels des connexions sont créées. Le prochain service rencontré dans la liste est le #4, qui ne nous intéresse pas ici. Ensuite, nous rencontrons le service #10 avec lequel on crée une connexion. Le prochain service rencontré, soit le #5, dessert la même paire O-D que SERV : dans ce cas, il est clair que les plages rencontrées jusque là soient celles vers D et E (destination des services #8, #9 et #10) sont terminées. Toutes les destinations plausibles n'ayant pas été rencontrées, nous poursuivons la descente pour rencontrer le service #11 faisant partie de la plage adjacente vers A et avec lequel on établit une connexion. La connexion avec le service #12 suivant n'est pas réalisée, car la plage adjacente vers D est déjà terminée. Le service suivant rencontré dessert la même paire O-D, nous terminons donc la plage adjacente vers A. Si l'on suppose que les destinations plausibles étaient A, D et E, celles-ci ayant été rencontrées, la plage de départ adjacente généralisée est terminée et donc le parcours s'arrête.

Par conséquent, le parcours dans la liste d'activités se poursuit jusqu'à la fin de la plage d'arrivée adjacente généralisée, c'est-à-dire jusqu'à ce qu'on est parcouru complètement les plages adjacentes de chacune des destinations plausibles. Toutefois, chaque plage ne se terminant en même temps, il est nécessaire de conserver son état (terminée ou non) de manière à ne générer des connexions qu'avec les plages adjacentes non encore terminées.

Il faut noter que rien de nouveau n'est ajouté au critère des plages horaires. La connexion s'effectue toujours avec les services de la plage de départ adjacente. C'est la réunion de tous les marchés sur un même schéma A/D qui semble compliquer le critère de connexion. La figure 4.9 présente séparément les schémas A/D des trois marchés considérés dans la figure 4.8.

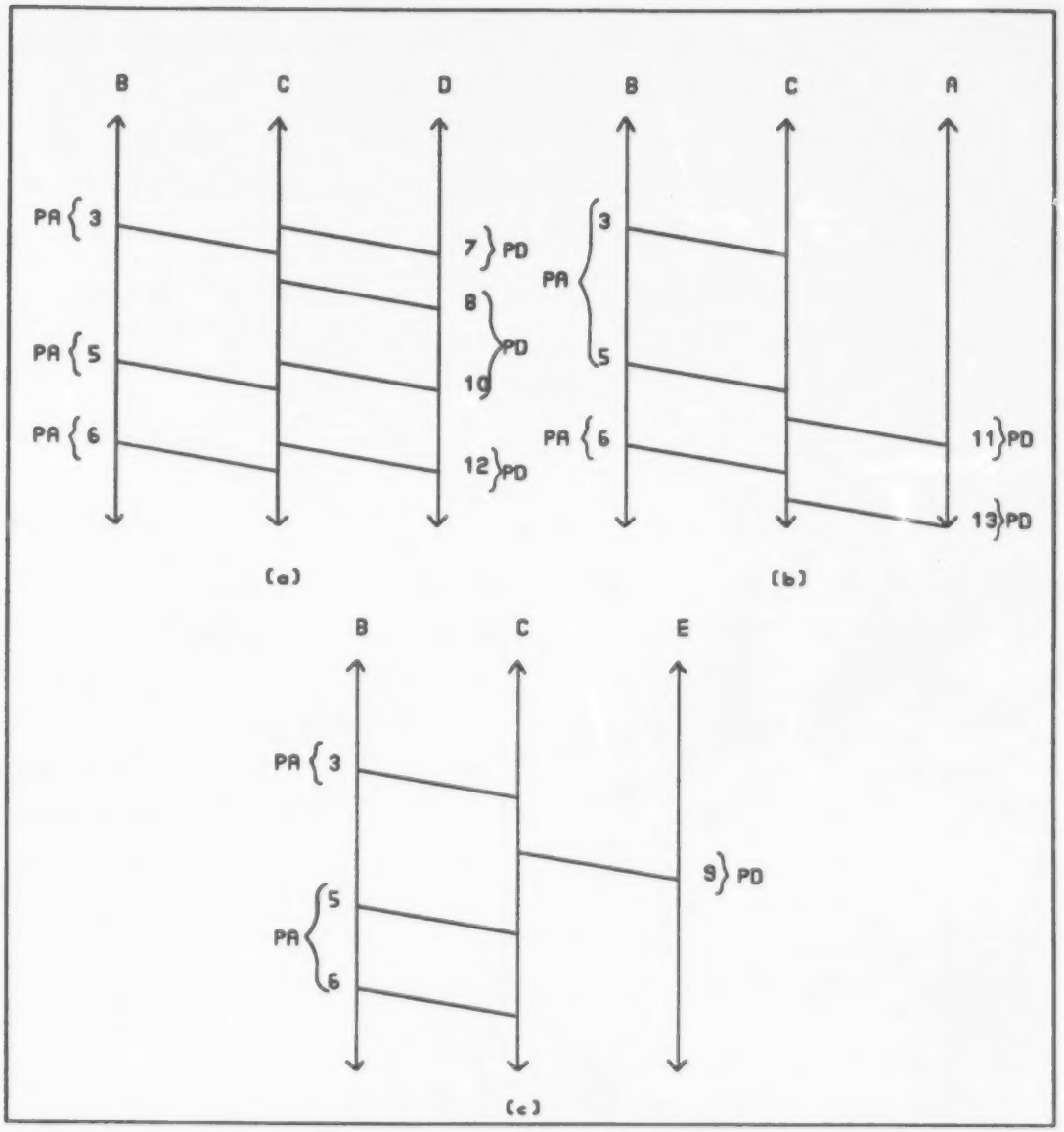


Figure 4.9 : Schémas A/D séparés

Notons en terminant que la plage d'arrivée contenant le service #3 varie selon la destination du marché considéré (cf. figure 4.9). Ceci est normal puisqu'une plage d'arrivée se termine quand la plage de départ adjacente commence. Les trois plages de départ commencant à des moments différents, la fin de la plage d'arrivée correspondante varie en conséquence.

Tout le traitement présenté ci-dessus pour générer les connexions à la ville d'arrivée du service ajouté, peut être transposé pour la génération des connexions à la ville de départ. Une simple permutation des termes "plages de départ" et "plage d'arrivée" ainsi que "descendre" et "monter" dans la liste est requise. Pour générer les services indirects à deux transferts, nous démontrons plus loin qu'aucun traitement supplémentaire sur les connexions n'est requis. Il s'agit en fait de combiner des services indirects à un transfert pour former ceux à deux transferts.

4.3.2.3 Présentation algorithmique

Cette section présente le détail de l'algorithme, avec toutes les étapes nécessaires à la construction des services indirects à un et deux transferts. La liste qui suit introduit les termes employés dans l'algorithme :

SERV : service ajouté;

NEXT : pointeur sur les différents services qui seront accédés pendant l'algorithme;

SIND1, SIND2 : pointeur sur les différents services indirects qui seront accédés pendant l'algorithme;

PLAGE : tableau d'indication de l'état des plages;

HMIN : heure minimale permise pour une connexion;

HMAX : heure maximale permise pour une connexion.

De plus la notation suivante est utilisée pour représenter un champ y d'un enregistrement x : x.y. Par exemple, SERV.HDEP représente l'heure de départ du service. Le lecteur est référé à l'appendice A pour la description détaillée des enregistrements utilisés.

De même, la notation T[i] représente la $i^{ième}$ case du tableau T. Nous avons aussi besoin de la proposition suivante :

Proposition 4.1

Un service indirect existe dans la banque de données si et seulement si la ou les connexions le formant ont été générées.

Nous présentons d'abord la génération des connexions à la ville d'arrivée du service. Les connexions à la ville de départ se produiront évidemment selon le même procédé. La figure 4.10 présente le déroulement global de cette première étape.

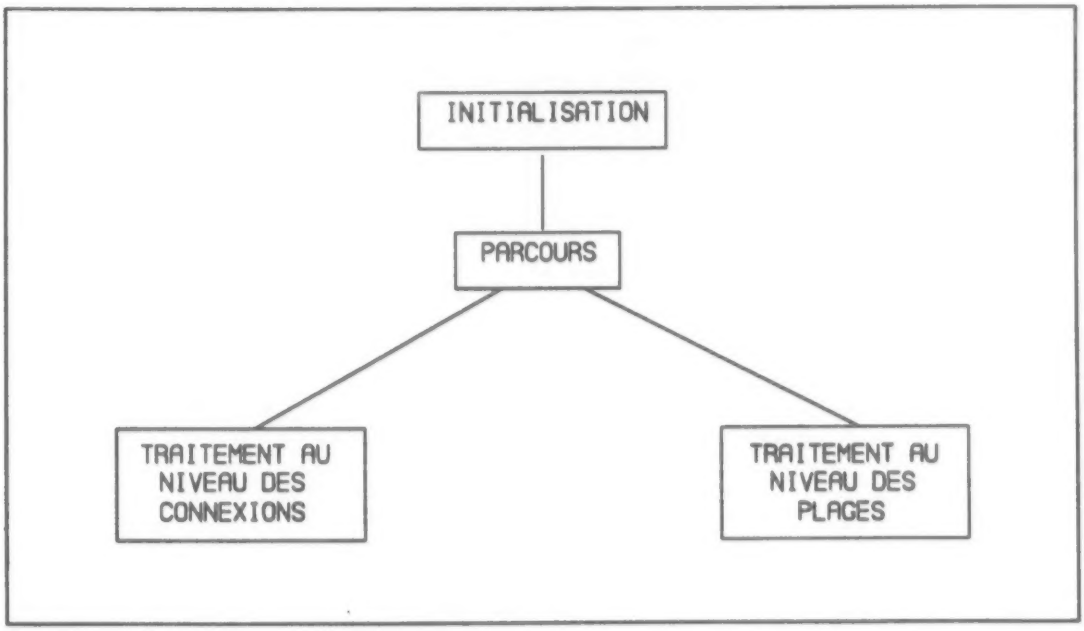


Figure 4.10 : Étapes du processus de génération

Initialisation

L'algorithme initialise les variables nécessaires à la descente dans la liste d'activités de la ville d'arrivée de SERV.

Pour conserver l'information sur l'état des plages horaires, nous utilisons un "tableau d'état de plage", nommé PLAGE. L'état "0" représente le cas où aucun service de cette plage n'a encore été rencontré. L'état "1" indique qu'au moins un service a été rencontré. La plage est terminée lorsque le tableau contient l'état "2".

Normalement, ce tableau serait initialisé à "0" pour toutes les villes. Toutefois, il est facile de remarquer une similarité entre la contrainte de plage et celle d'itinéraire plausible (cf. section 4.1.1). Dans le premier cas, il faut indiquer pour quelles destinations la plage adjacente généralisée n'est pas terminée, tandis que dans le deuxième cas l'itinéraire étant déjà spécifié en partie par le service SERV, il ne reste qu'à spécifier les destinations plausibles.

L'état de ces deux contraintes peu tenir dans un seul tableau de ville, soit celui des plages. Celui-ci est initialisé de sorte que les villes correspondant à des destinations non plausibles contiennent l'état "2". Ainsi aucune connexion ne sera créée sur ces itinéraires puisque les plages correspondantes seront considérées dès le début comme terminées. Le pseudocode suivant présente cette initialisation :

```
Pour chaque ville v faire
  si l'itinéraire défini par (SERV.VDEP-SERV.VARR-V)
  est plausible alors
      PLAGE[V]← "0"
  sinon
      PLAGE[V]← "2".
```

Pour les contraintes de délai de connexion, l'algorithme initialise deux variables définissant les heures minimale et maximale entre lesquelles une connexion est acceptée :

```
HMIN ← SERV.HARR + DELAIMIN
HMAX ← SERV.HARR + DELAIMAX.
```

Les variables DELAIMIN et DELAIMAX contiennent les délais de connexion minimaux et maximaux. Ainsi un délai minimum de DELAIMIN minutes est requis pour établir une connexion, mais une attente de plus de DELAIMAX minutes est jugée inacceptable. Ces délais sont fixés pour chaque mode de transport.

Parcours

Il s'agit de générer les connexions tout au long de la descente dans la liste d'activités de la ville d'arrivée du service SERV. Le parcours s'effectue comme suit :

```
NEXT ← SERV.JNEXT
tant que NEXT existe faire
    si activité = départ alors
            TRAITEMENT AU NIVEAU DES CONNEXIONS
    sinon
            TRAITEMENT AU NIVEAU DES PLAGES.
```

Traitement au niveau des connexions

L'algorithme vérifie si la limite maximale de délai de connexion est atteinte :

```
si NEXT.HDEP > HMAX alors
   STOP.
```

Il vérifie ensuite si le service NEXT appartient à une plage adjacente terminée ou non terminée. Si elle est terminée, il passe au prochain service :

```
si PLAGE [NEXT.VARR] = "2" alors
   NEXT ← NEXT.INEXT.
```

Si la plage n'est pas terminée et que le délai minimal de connexion est respecté, l'algorithme crée le service indirect correspondant à la connexion SERV+NEXT :

si NEXT.HDEP > HMIN alors

- *création du service indirect*
- *insertion dans la banque de données*
- ***génération des services indirects à deux transferts avec SERV à gauche.***

Finalement, l'algorithme indique que la plage de cette destination a été rencontrée, puis on passe au prochain service dans la descente :

PLAGE [NEXT.VARR] ← "1"

NEXT ← NEXT.INEXT.

Génération des services indirects à deux transferts avec SERV à gauche

Il s'agit de prolonger vers la droite le service indirect nouvellement créé, soit (SERV+NEXT).

Supposons qu'un service indirect à deux transferts (SERV+NEXT+AUTRE) existe dans la banque de données. Par la proposition 4.1, nous pouvons affirmer que les connexions entre SERV et NEXT de même qu'entre NEXT et AUTRE ont été générées. Si la connexion entre NEXT et AUTRE a été générée, alors la proposition 4.1 permet d'affirmer que le service indirect (NEXT+-AUTRE) existe; de même pour le service indirect (SERV+NEXT).

Par conséquent, pour former les services indirects de la forme (SERV+NEXT+AUTRE) où AUTRE est quelconque, il s'agit d'accéder aux services indirects de la forme (NEXT+AUTRE) et de les "combiner" avec le service indirect (SERV+NEXT). La figure 4.11 présente un exemple de ce type de combinaison. Aucune connexion n'est générée, il s'agit simplement de tester la plausibilité de l'itinéraire formé.

L'algorithme vérifie d'abord s'il existe des itinéraires plausibles :

S'il existe une ville v tel que l'itinéraire
défini par (SERV.VDEP-SERV.VARR-NEXT.VARR-V)
est plausible alors
- continue
sinon
- retour.

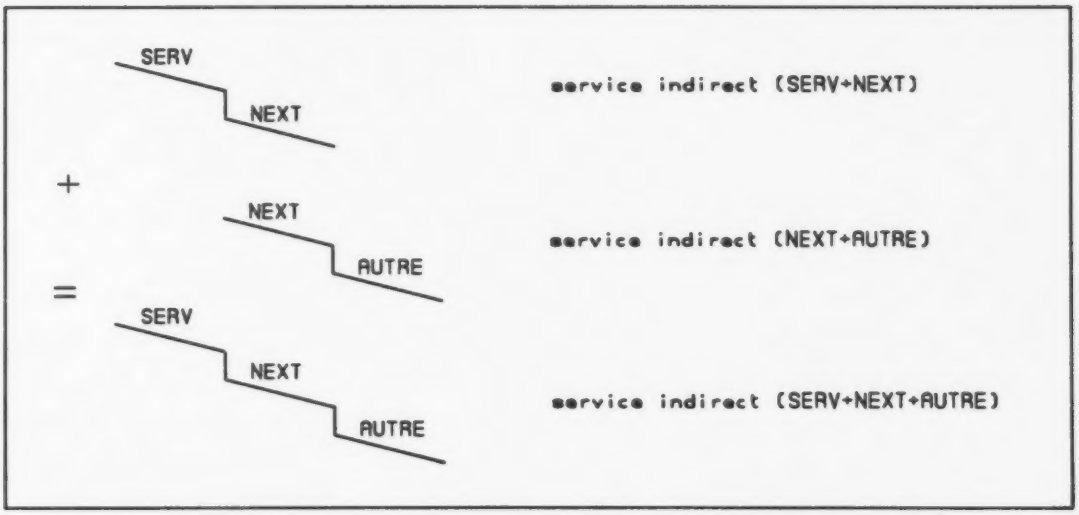


Figure 4.11 : Combinaison de services indirects

Ensuite, il faut parcourir les services indirects à un transfert avec NEXT comme premier segment. Ceux-ci sont directement accessibles par le pointeur GAUCHE du service.

SIND ← NEXT.GAUCHE
tant que SIND existe faire
si l'itinéraire défini par (SERV.VDEP-SERV.VARR-NEXT.VARR-SIND.VARR) est plausible alors
- création et insertion du service indirect
SIND ← SIND.NEXT1.

Traitement au niveau des plages

Si l'activité rencontrée lors de la descente est une arrivée et qu'elle provient de la même origine que SERV, une mise à jour de la validité des plages adjacentes est effectuée. C'est-à-dire qu'on indique l'état terminé "2" aux plages adjacentes déjà rencontrées :

si NEXT.VDEP = SERV.VDEP alors
pour chaque ville v faire
si PLAGE[v] = "1" alors
PLAGE[v] ← "2".

La descente ne se poursuit que s'il reste des plages adjacentes non rencontrées, i.e. si la plage de départ adjacente généralisée n'est pas terminée :

pour chaque ville v faire
si PLAGE[v] = "0" alors
- on continue.

Lorsque toutes ces étapes de génération des connexions ont été réalisées à la ville d'arrivée, de même qu'à la ville de départ du service, il ne reste qu'à générer les services indirects à deux transferts où SERV est au centre.

Considérons un service indirect à deux transferts, soit SIND, où le service SERV forme le deuxième segment, i.e. celui du centre. Si SIND a la forme (SERV1+SERV+SERV2) et existe dans la banque de données, la proposition 4.1 nous permet de dire que les services indirects (SERV1+SERV) et (SERV+SERV2) existent aussi.

Tout service indirect SIND est donc formé de la combinaison de deux services indirects à un transfert, où SERV constitue le premier segment de l'un et le deuxième de l'autre. Les services indirects à un transfert utilisant SERV de cette façon ont déjà été créés aux étapes précédentes. Il ne

s'agit donc que d'accéder ceux-ci et de les combiner selon le même principe que celui montré à la figure 4.11. Aucune nouvelle connexion n'est nécessaire, il faut seulement vérifier la plausibilité de l'itinéraire défini par l'union des deux services indirects.

Les pointeurs GAUCHE et DROITE du service SERV permettent d'accéder directement aux listes de services indirects recherchés. La génération des services indirects à deux transferts avec SERV au centre s'effectue donc selon l'algorithme suivant :

SIND ← SERV.D[illegible]
tant que SIND existe faire
s'il existe une ville v telle que l'itinéraire
défini par (SIND.VDEP-SERV.VDEP-SERV.VARR-V)
est plausible alors
SIND2 ← SERV.GAUCHE
tant que SIND2 existe faire
si l'itinéraire défini par (SIND.VDEP-
SIND2.VDEP-SIND.VARR-SIND2.VARR) est
plausible alors
- création du service indirect
- insertion dans la banque de données
SIND2 ← SIND2.NEXT1
SIND ← SIND.NEXT2.

5. RÉALISATION DU LOGICIEL

Au cours de ce chapitre, nous traitons de l'implantation informatique des composantes du simulateur JET-SET/3 développées au cours de ce stage. Nous présentons d'abord les procédures de mise à jour de la banque de données, puis nous discutons des procédures reliées à la gestion et à la présentation des interfaces, et notamment de leurs aspects graphiques.

Le langage C a été choisi pour l'implantation informatique du logiciel. Ses qualités de portabilité et d'efficacité ont prévalu dans cette décision. Le simulateur est présentement développé sur un micro-ordinateur IBM AT avec écran couleur EGA et système d'exploitation DOS. Des implantations sur des micro-ordinateurs PS-2 et COMPACT, ainsi que sur des stations SUN sont envisagées. Il faut donc s'assurer que les programmes s'adapteront aisément à des exécutions sous OS-2 et UNIX.

Le language C offre toutes les composantes nécessaires à la programmation de systèmes évolués, tout en imposant peu de restrictions quant à l'utilisation. Il permet ainsi d'obtenir un maximum d'efficacité. De plus, ce langage permet l'allocation dynamique de mémoire, ce qui est très important pour notre application puisque la banque de données est située en mémoire et non sur disque.

5.1 Gestion de la banque de données

Le tableau I résume les principaux modules de gestion de la banque de données. Les sections suivantes présentent les fonctions principales de ces modules.

Tableau I : Modules de gestion de la banque de données

1. Module d'initialisation
2. Module d'insertion
3. Module d'enlèvement
4. Module de connexion
5. Module charger/sauver

5.1.1 Module d'initialisation

Ce module assure l'initialisation de la structure de la banque de données avant le chargement des données. Cette étape préliminaire consiste à initialiser toutes les structures d'accès aux données (matrice O-D, tableau des villes, tableau des véhicules, ...) avec des pointeurs nuls, ainsi qu'à construire la structure contenant les villes de transferts plausibles pour chaque marché.

L'algorithme de génération des connexions présenté au chapitre 4 nécessite de déterminer rapidement si une ville donnée est considérée comme un transfert plausible pour un service indirect à un transfert sur un marché donné. De même pour créer des services indirects à deux transferts, l'algorithme doit déterminer si une paire de villes donnée est considérée comme une paire de transferts plausible pour un service indirect à deux transferts sur un marché donné.

Pour satisfaire ce type de requêtes la structure développée est la suivante. Pour les transferts simples, nous utilisons un tableau de villes où chaque case représente une ville de transfert et est constituée d'un pointeur sur une "matrice de marchés". Cette matrice est une matrice booléenne où chaque case correspond à un marché et contient "1" si la ville considérée est un transfert plausible pour ce marché et "0" sinon. Pour les transferts doubles, nous utilisons une matrice de villes, où chaque case représente une paire de villes de transfert et est constituée d'un pointeur sur une matrice de marchés telle que décrite ci-haut. Ainsi pour savoir si

une ville donnée, soit A, est un transfert plausible pour un marché donné, soit C-G, on accède à la première case du tableau de villes qui nous renvoit à une matrice de marchés dans lequelle on vérifie si la case correspondante au marché C-G contient "1".

Le chargement de ces structures s'effectue en parcourant l'ensemble des itinéraires indirects plausibles pour desservir les marchés; celui-ci est emmagasiné sur disque. Chaque itinéraire est représenté par un triplet de la forme (une ou deux villes de transfert, l'origine, la destination). Pour chaque triplet rencontré, la matrice de marchés du(des) transfert(s) spécifié(s) est accédée et la case correspondant au marché défini par les deux dernières coordonnées du triplet est marquée d'un "1". Si le transfert est rencontré pour la première fois, l'espace pour la matrice de marchés doit être préalablement alloué.

5.1.2 Module d'insertion

Ce module assure la mise à jour de la banque de données lorsqu'un service est ajouté. Avant d'effectuer l'insertion, l'enregistrement du nouveau service doit être alloué et complété avec les informations fournies par l'usager (cf. Appendice A, pour détail de l'enregistrement). Par la suite, nous procédons à l'insertion du service dans la structure de multiliste décrite au chapitre 3.

Le service est d'abord inséré dans la liste de l'horaire de service de la paire de villes correspondante. Cette insertion est réalisée de façon très classique en utilisant un algorithme standard d'insertion dans une liste doublement chaînée et ordonnée (cf. Appendice B, fct 1).

Ensuite nous insérons le service dans la liste de la route-horaire du véhicule effectuant ce service. L'insertion s'effectue selon le même algorithme.

Nous procédons par la suite à l'insertion du service dans la liste d'activités de sa ville de départ et dans celle de sa ville d'arrivée. Ici,

le positionnement préalable à l'insertion ne débute pas en tête de liste. Nous profitons plutôt de l'organisation de multiliste pour obtenir un point d'entrée dans les listes d'activités, qui soit plus près de la position recherchée. À cette fin, nous utilisons le service précédent ou, s'il n'existe pas, le service suivant de la liste de l'horaire de service. Ceux-ci sont directement accessibles par les pointeurs *PREC* et *NEXT* de l'enregistrement du service. Ce service précédent ou suivant fait partie des mêmes listes d'activités que le nouveau service. Étant donné sa proximité probable au niveau des heures d'arrivée et de départ, nous pouvons supposer qu'il offre un bon point d'entrée dans ces listes d'activités.

Cette astuce n'est toutefois possible que s'il existe un service précédent ou suivant dans l'horaire de service. Autrement, la recherche débute en tête de liste.

Après ce positionnement initial, il faut rechercher la position finale du nouveau service dans la liste d'activités en effectuant les comparaisons avec l'heure de l'activité à la ville, i.e. parfois avec l'heure d'arrivée, parfois avec l'heure de départ des services rencontrées. De la même façon, les pointeurs à suivre dépendent de l'activité, soient *IPREC* et *INEXT* pour un départ, sinon *JPREC* et *JNEXT*.

Tel qu'illustré au chapitre 4, l'insertion d'un service dans les listes d'activités peut briser l'état courant des plages et occasionner ainsi l'invalidité de services indirects préalablement construits. Ce bris de plages ne survient que si le nouveau service forme une plage à lui seul, scindant ainsi en deux parties la plage opposée.

Pour établir s'il y a ou non bris de plages, il faut parcourir les listes d'activités de façon à déterminer les plages adjacentes s'étalant de part et d'autre du nouveau service. Si une plage se retrouve avant et après, c'est qu'il y a bris de cette plage. La fonction 2 de l'appendice B présente le détail de cette procédure de vérification.

Lorsqu'une plage adjacente est effectivement brisée, certains services indirects provenant d'anciennes connexions doivent être éliminés, car ils ne respectent pas la nouvelle disposition des plages (cf. section 4.3.1). L'enlèvement de ces services indirects est réalisé en éliminant chacun d'eux de toutes les listes auxquelles il appartient. Cette étape est facilitée par le double chaînage existant dans la structure de multiliste de la banque de données.

La mise à jour complète de la banque de données, suite à l'insertion d'un nouveau service, requiert évidemment la création des services indirects résultant de connexions générées avec le nouveau service. Le module de connexion (cf. section 5.1.4) est consacré à ce processus de génération des services indirects.

5.1.3 Module d'enlèvement

Ce module assure la mise à jour de la banque de données lors de l'extraction d'un service. Le service est d'abord enlevé de chacune des listes auxquelles il appartient, soient celle de son horaire de service, celle de sa route-horaire et finalement celles de ses listes d'activités. Le positionnement sur le service, préalable à toutes ces éliminations, est facilité par le côté interactif du système. En effet, le service à enlever étant toujours un service identifié à l'écran, on se sert des pointeurs de gestion des fenêtres pour accéder rapidement au service visé. Par la suite, de simples manipulations de pointeurs suffisent pour mettre à jour l'ensemble des listes.

Tous les services indirects utilisant le service éliminé doivent aussi être retirés de la structure de multiliste. Rappelons que ces services indirects sont répartis sur trois ensembles accessibles directement par les pointeurs *GAUCHE*, *DROITE* et *DEUX* du service (cf. chapitre 3, section 3.2.3.5). Par conséquent, l'élimination s'effectue en

parcourant la liste de chacun des trois ensembles, où chaque service indirect rencontré est enlevé de chacune des listes auxquelles il appartient.

Tout comme l'insertion, l'enlèvement d'un service peut modifier la disposition des plages, en suscitant l'union de plages adjacentes (cf. chapitre 4, section 4.3.1). Cette union ne survient toutefois que si le service enlevé formait auparavant une plage à lui seul, son retrait provoquant nécessairement la réunion des plages adjacentes de part et d'autre.

Pour établir s'il y a ou non union de plages, il faut parcourir les listes d'activités de façon à déterminer les plages adjacentes s'étalant de part et d'autre du service enlevé. Si une plage se retrouve avant et après le service enlevé, c'est qu'il y a union de ces plages. L'algorithme présenté pour déterminer les bris de plages utilisait le même procédé; il peut donc être utilisé pour le cas présent.

Lorsque deux plages sont effectivement unies, les services de l'ancienne plage adjacente du service éliminé doivent être connectés avec les services de leur nouvelle plage adjacente (cf. chapitre 4, section 4.3.1). Ainsi pour chaque service de cette ancienne plage adjacente or appelle l'algorithme de génération des connexions présenté au chapitre 4, mais en spécifiant certains paramètres de telle sorte que le service soit connecté seulement avec les services de la nouvelle plage adjacente citée ci-haut.

5.1.4 Module de connexion

Ce module s'occupe de la création de services indirects résultants de l'insertion d'un nouveau service dans la banque de données. La première étape consiste à générer les connexions valides entre services; elle a été présentée au chapitre 4, sous le vocable "algorithme de génération des connexions".

Pour chaque connexion générée, ce module crée un service indirect et l'insère dans la banque de données. Cette partie est très simple, elle ne requiert que quelques insertions dans des listes doublement chaînées.

5.1.5 Module charger/sauver

Ce module s'occupe de "sauver" sur disque une banque de données couramment en mémoire ou de charger en mémoire une banque de données emmagasinée sur disque.

L'objectif premier de la méthode employée est la rapidité d'opération, car le système est interactif. Par conséquent, il faut éviter de parcourir les services plusieurs fois. La technique que nous avons développée permet le chargement en une seule "passe", c'est-à-dire avec un seul parcours de tous les services. La sauvegarde, elle, s'effectue en deux passes. Cependant, une de celles-ci, la numérotation, sert pour les deux opérations, soient sauver et charger.

La méthode repose sur une numérotation préalable des services et services indirects pour ensuite remplacer chaque pointeur par le numéro du service ou service indirect qu'il désigne. Ainsi tous les champs pointant vers des services ou des services indirects contiennent des entiers pendant leur séjour sur disque. De cette façon, nous retenons les liens virtuels existants entre les données de la banque.

Sauvegarde

La première étape consiste en un parcours séquentiel des services et services indirects pour les numéroter. Par la suite, l'algorithme parcourt à nouveau tous les services en effectuant deux opérations pour chacun d'eux. La première opération consiste à remplacer chaque pointeur par le numéro du service qu'il désigne. La deuxième opération s'occupe d'écrire le service dans le fichier des services. Cette étape est aussi effectuée pour tous les services indirects.

La prochaine étape de l'algorithme concerne la sauvegarde de chacune des structures d'accès aux données, i.e. aux services et services indirects. Ici aussi nous procédons d'accès d'abord à un remplacement des pointeurs par les numéros des services et services indirects qu'ils désignent, puis à l'inscription dans un fichier.

La banque de données ainsi stockée se répartit sur six fichiers. La description de ces fichiers est présentée au tableau II.

Tableau II : Structure des fichiers

Nom du fichier	Description
service.dir	- le nombre d'enregistrement - la suite des enregistrements de services
service.ind	- le nombre d'enregistrement - la suite des enregistrements de services indirects
matrice.dir	- la suite des cases de la matrice O-D
activité.dir	- la suite des cases du tableau de villes
route.hor	- la suite des cases du tableau de véhicules
index.ind	- suite des index pour chaque marché. Chaque index est représenté par le nombre d'indice et la suite des indices

Chargement

Après l'ouverture des fichiers, la première opération consiste à allouer l'espace mémoire pour tous les services et les services indirects de la banque de données :

Étape 0. Recherche du nombre de services

```
*NB = getw(fp_serv);
```

Étape 1. Allocation des pointeurs de services temporaires

```
ptr_dir = (struct serv_dir*)malloc(*NB*sizeof(struct serv_dir*));
```

Étape 2. Allocation des services

```
for (i=0;i <=*NB-1;i++)
    ptr_dir[i] = (struct serv_dir*)malloc(sizeof(struct serv_dir));
```

À la fin de cette première étape, les tableaux PTR_DIR et PRT_IND contiennent les adresses de toutes les données ultérieurement chargées.

Nous procédons ensuite au chargement des services. Chaque service est chargé dans l'espace mémoire qui lui est déjà assigné, soit l'adresse PTR_DIR[i] pour le service #i :

```
for (i=0;i < NB_DIR-1;i++){
     serv = ptr_dir[i];
     fread(serv,sizeof(struct serv_dir),1,fp_serv);
     .
     .
     .
}.
```

À chaque chargement d'un service, nous ajustons les pointeurs de celui-ci, de façon à ce qu'ils détiennent l'adresse des services qu'ils désignent. L'adresse du service #i se trouvant dans PTR_DIR i , cet ajustement s'effectue aisément :

```
serv.next = ptr_dir[(long)serv.next]
serv.prec = ptr_dir[(long)serv.prec]
.
.
.
```

Nous effectuons le chargement des services indirects selon le même procédé.

Finalement, les différentes structures d'accès aux services et services indirects (matrice de ville, index, tableau de véhicules, ...) sont chargées en mémoire en ajustant ici aussi les champs pointeurs avec les vraies adresses.

5.2 Gestion des interfaces

5.2.1 Outils logiciels

L'implantation des interfaces du simulateur a été réalisée à l'aide d'un "package" de gestion de fenêtres nommé PANEL PLUS (cf. Panel Plus (1987)). Ce "package" est portable, modifiable et permet un traitement de fenêtres en mode texte et en mode graphique. Son utilisation a permis d'accélérer le développement du simulateur en nous évitant de concevoir notre propre ensemble de procédures pour la gestion des fenêtres.

PANEL PLUS est formé de deux parties. La première comprend un ensemble de programmes permettant la construction interactive de l'écran désiré ainsi que la génération des structures correspondantes. L'autre partie consiste en une librairie de fonctions (code fourni) pour la manipulation des écrans et fenêtres, en mode texte ou graphique. La disponibilité du code de la librairie offre la possibilité de modifier les fonctions pour les adapter à nos besoins.

De plus, PANEL PLUS n'est pas contraignant et laisse toute liberté à l'usager quant à l'utilisation de ses fonctions. La consultation du code de ces fonctions est une source riche d'exemples pour la conception de nouvelles fonctions.

D'autres choix de logiciels de gestion de fenêtres et de graphisme ont été analysés. Quoique possédant de grands avantages, notre analyse nous a toutefois amené à conclure qu'ils auraient tous entraînés un ou plusieurs des inconvénients suivants : des coûts élevés, des adaptations nombreuses, un manque de souplesse, un manque d'efficacité ou une longue période d'apprentissage. PANEL PLUS, qui était disponible au Centre de recherche sur les transports (lieu du stage), s'est donc avéré être le meilleur choix pour les travaux que nous voulions entreprendre, et de fait il a réellement permis de réduire le temps de développement prévu.

PANEL PLUS permet l'exécution de programmes en mode graphique, mais il n'inclut aucune fonction de graphisme. Nous avons donc eu besoin d'un autre outil pour réaliser les représentations graphiques. Celles-ci étant relativement simples, les fonctions graphiques intégrées à la dernière version du compilateur C de Microsoft sont apparues suffisantes.

Plusieurs modules de bas niveau sont inclus dans le "package" PANEL PLUS, ils permettent son utilisation avec différents "packages" graphiques. La librairie graphique de Microsoft C ne faisant pas partie des choix offerts, il a fallu créer un nouveau module de bas niveau. Plus précisément, il s'agissait d'intégrer à un module de base des fonctions pour l'utilisation du mode graphique, pour la gestion du curseur, pour l'affichage de texte, pour sauver et rétablir une partie d'écran, de même que pour gérer les attributs tels les couleurs du fond, du texte, ... Pour optimiser la gestion des écrans, nous avons aussi intégré à notre module de bas niveau la possibilité de choisir entre un mode graphique ou texte pour l'affichage. Ceci a été réalisé en combinant le nouveau module avec celui de base (mode texte) utilisé par PANEL PLUS. Une variable externe permet pendant l'exécution de passer du mode texte au mode graphique et inversement.

5.2.2 Département Planification

Le tableau III résume les différents modules gérant l'interface du département Planification. Les sections suivantes présentent une description sommaire des fonctions principales de chacun des modules. Ces descriptions sont du niveau de l'implantation informatique. Pour une analyse plus conceptuelle, le lecteur doit se référer au chapitre 2, section 2.3.1.

Tableau III : Modules du département Planification

1. Module d'interception
2. Module de positionnement
3. Module de défilement
4. Module de changement de liste
5. Module d'insertion
6. Module d'enlèvement
7. Module de modification
8. Module de mise à jour

5.2.2.1 Module d'interception

Ce module s'occupe de capter les entrées de l'usager et d'activer les fonctions correspondantes, et le cas échéant d'envoyer un signal d'erreur. Les entrées proviennent du clavier et de la souris.

5.2.2.2 Module de positionnement

Lorsqu'un participant décide de changer de fenêtre ou de zone, ce module s'occupe de mettre à jour l'écran et les variables d'environnement identifiant la position de l'usager dans l'interface. Cette position est conservée dans deux variables d'environnement, soient NOZONE et NOFEN, contenant respectivement le numéro de la zone active (0,1) et le numéro de la fenêtre courante (0 à 3). Les autres modules de ce département utilisent ces variables pour orienter leurs opérations vers la bonne fenêtre.

5.2.2.3 Module de défilement

Ce module s'occupe de faire défiler le contenu des fenêtres. Les listes contenues dans ces fenêtres ne sont pas emmagasinées dans des fichiers, mais sont la représentation dynamique des informations courantes de la banque de données. Par conséquent, chaque fenêtre doit posséder un pointeur de position désignant la première donnée, service ou service indirect, affichée dans la fenêtre.

La fonction de défilement s'occupe de déplacer ces pointeurs de fenêtres de façon à avancer ou reculer dans les listes doublement chaînées des horaires de service ou publiés. Ce défilement d'une fenêtre s'effectue en trois étapes. Premièrement, déplacement du pointeur de la fenêtre courante conséquemment à l'opération désirée. Puis, chargement de la zone de données de la fenêtre par un parcours de la liste doublement chaînée de l'horaire; et finalement, réaffichage de cette fenêtre.

5.2.2.4 Module de changement de liste

Les fonctions de ce module permettent à l'usager de remplacer l'horaire de la fenêtre courante par l'horaire d'une autre paire de villes. Cette manipulation s'effectue en plusieurs étapes. Le programme incite d'abord l'usager à spécifier la nouvelle paire de villes dans les cases d'origine et de destination de la fenêtre. À l'aide de cette information, le pointeur de la fenêtre est modifié de façon à désigner le premier service ou service indirect de la liste représentant l'horaire de la nouvelle paire de villes. Puis, la zone de données de la fenêtre est mise à jour et celle-ci est réaffichée.

5.2.2.5 Module d'insertion

Ce module permet l'insertion d'un nouveau service dans l'horaire de service de la fenêtre courante. L'usager spécifie l'heure de départ de ce nouveau service, avec laquelle le système calcule l'heure d'arrivée. Le service correspondant est créé puis envoyé à la procédure d'insertion dans la banque de données. La mise à jour des fenêtres, suite à cette insertion, est réalisée par le module de mise à jour (cf. section 5.2.2.8).

5.2.2.6 Module d'enlèvement

Ce module assure l'enlèvement d'un service désigné par la position dans la fenêtre courante. La recherche du service dans la banque de données

est accélérée par l'utilisation du pointeur de fenêtre comme point d'entrée dans la liste de l'horaire de service de la paire de villes considérée. Le parcours à effectuer est ainsi réduit à une longueur égale à la position dans la fenêtre. Par la suite, le service est "passé" à la procédure assurant l'élimination du service de la banque de données. Ici aussi le module de mise à jour (cf. section 5.2.2.8) s'assure de mettre à jour les fenêtres affectées par la modification.

5.2.2.7 Module de modification

Ce module ne fait qu'appeler le module d'enlèvement, puis celui d'insertion. Il donne l'illusion d'une modification quand, en réalité, le service désigné est remplacé par un nouveau service.

5.2.2.8 Module de mise à jour

Ce module sert pour toutes les opérations modifiant l'état de la banque de données. Avant une telle opération, il conserve les informations nécessaires sur les pointeurs de fenêtres, de façon à ce que ces derniers puissent être recouvrés si l'opération effectuée les rend invalides (i.e. si le service désigné est éliminé). Après l'opération, ce module met à jour le contenu des fenêtres, à l'aide des informations conservées et de l'état actuel de la banque de données.

5.2.3 Département Gestion des opérations

Le tableau IV présente les différents modules gérant l'interface de ce département. Les sections suivantes présentent brièvement les principales fonctions de chacun des modules.

Tableau IV : Modules du département Gestion des opérations

1. Module du graphisme
2. Module d'interception
3. Module d'inspection
4. Module de défilement
5. Module de déplacement
6. Module d'insertion
7. Module d'enlèvement
8. Module de croisement
9. Gestion de l'horaire
10. Gestion du graphisme ESP-TPS

5.2.3.1 Module du graphisme

Ce module regroupe toutes les fonctions ayant trait à l'affichage graphique. Ce regroupement permet d'isoler l'aspect graphique, de façon à ce qu'un changement de librairie graphique soit facilité si le besoin s'en fait sentir. Ce module comprend les fonctions pour tracer les axes des diagrammes, ainsi que les deux types de représentation des routes-horaires.

Le diagramme véhicule-temps (cf. chapitre 2, section 2.3.2) de l'interface est réalisé premièrement par un tracé des axes, puis en dessinant chacune des routes-horaires des véhicules de l'axe vertical. Ce deuxième tracé est effectué en parcourant la liste correspondant à chaque route-horaire. Chaque service est représenté par une oscillation rectangulaire (⎍⎍) de base proportionelle au temps de déplacement, et est positionné en abscisse par rapport à son heure de départ et en ordonnée selon le numéro du véhicule. L'appendice B contient le code de cette procédure de représentation graphique d'une route-horaire (cf. fct 5).

Pour obtenir le diagramme véhicule-temps en entier, la procédure de traçage de route-horaire doit être appelée pour chaque véhicule présenté sur l'axe vertical.

Le diagramme espace-temps, offert comme outil d'aide à la décision, est créé de façon semblable : tracé des axes d'abord, puis affichage de chaque route. L'algorithme pour ce second tracé de route repose aussi sur un parcours en parallèle de la liste représentant la route-horaire. Pour plus de détails, l'appendice B contient le code de cette seconde procédure graphique de traçage des routes-horaires (cf. fct 4).

5.2.3.2 Module d'interception

Ce module intercepte les entrées (clavier et souris) de l'usager et active la fonction correspondante, et le cas échéant envoie un signal d'erreur.

5.2.3.3 Module d'inspection

Ce module sert à obtenir des informations précises sur un service affiché dans une route-horaire du diagramme véhicule-temps. Le module est activé lorsque l'usager pointe avec la souris un point de la surface délimitée par les deux axes du graphique. Les coordonnées du point envoyées par la souris sont des coordonnées "pixel" (physique). Pour déterminer le service désigné, il faut transformer ces coordonnées "pixel" en coordonnées du diagramme véhicule-temps, c'est-à-dire en coordonnées (véhicule, heure). Cette transformation est réalisée par un calcul utilisant la position et l'échelle des axes du diagramme. La fonction 3 de l'appendice B présente le détail de cette transformation.

Avec les deux informations fournies par les nouvelles coordonnées, soient le numéro de véhicule et l'heure, le programme parcourt la liste correspondant à la route-horaire du véhicule à la recherche d'un service où la coordonnée heure se situe entre l'heure de départ et l'heure d'arrivée du service. Si un tel service est trouvé, la recherche est arrêtée et les données de ce service sont affichées dans une fenêtre positionnée sur le point indiqué par la souris.

5.2.3.4 Module de défilement

Ce module permet de faire défiler l'axe vertical du diagramme véhicule-temps de manière à faire afficher les routes-horaires de d'autres véhicules.

Le numéro du premier véhicule de l'axe est conservé dans la variable NOFIRST. Le programme modifie le contenu de cette variable du pas de déplacement désiré par l'usager. La procédure pour tracer le graphique utilise la variable NOFIRST, aussi un simple appel à cette fonction permet d'afficher le nouveau contenu du diagramme. Cet affichage étant suffisamment rapide, l'effet est comparable à un défilement de liste numérique.

5.2.3.5 Module de déplacement

Ce module s'occupe des opérations nécessaires pour déplacer un ou plusieurs services à l'intérieur d'une route affichée à l'écran.

Trois étapes sont nécessaires pour effectuer un déplacement. La première concerne la localisation des services affectés; l'usager ayant indiqué avec la souris la section de route visée, il faut transformer les coordonnées "pixel" reçues en coordonnées (véhicule-heure) (cf. Appendice B, fct 3), puis parcourir la liste de la route-horaire correspondante, à la recherche des services affectés. La deuxième étape consiste d'abord à calculer le déplacement occasionné par la nouvelle heure de départ spécifiée par l'usager (simple soustraction entre la nouvelle heure et l'heure actuelle). L'intervalle de temps ainsi obtenu est conservé dans la variable DIFF_HRE. Par la suite, le programme parcourt les services affectés et pour chacun effectue d'abord une élimination de la banque de données, puis une nouvelle insertion. Le service ajouté possède les mêmes caractéristiques que le service correspondant enlevé, sauf pour les heures d'arrivée

et de départ auxquelles on additionne DIFF_HRE. Après avoir déplacé ainsi chaque service, la route est réaffichée par la procédure graphique déjà présentée (cf. appendice B, fct 5).

5.2.3.6 Module d'insertion

Ce module réalise les deux genres d'insertions offertes à l'usager, soient la manuelle et celle par copie. Pour l'insertion manuelle, l'usager doit spécifier les données du service à insérer. Avec les informations obtenues, le système crée un enregistrement pour le nouveau service et appelle la procédure d'insertion dans la banque de données. Par la suite, le programme appelle la fonction graphique pour réafficher la route-horaire ayant reçu le nouveau service.

L'insertion par copie se réalise en deux étapes. La première consiste en la localisation de la section de route-horaire à copier. Cette section est désignée par l'usager avec la souris et il faut donc transformer les coordonnées "pixel" reçues en coordonnées (véhicule, heure) (cf. Appendice B, fct 3), puis effectuer la recherche des services à copier en parcourant la liste de la route-horaire du véhicule spécifié par les coordonnées. La deuxième étape nécessite d'abord la spécification de la route-horaire où l'on veut réaliser l'insertion. Ici aussi nous devons d'abord transformer les coordonnées "pixel" reçues de l'usager en coordonnées (véhicule, heure). Avec le numéro du véhicule ainsi trouvé, il s'agit de créer, pour chaque service de la section de route à copier, un service semblable se différenciant seulement par le numéro de véhicule, puis d'appeler la procédure d'insertion dans la banque de données. À la fin de ces deux étapes, la route du véhicule est réaffichée, exhibant l'insertion.

5.2.3.7 Module d'enlèvement

Ce module exécute l'enlèvement d'ou ou plusieurs services d'une route-horaire en deux étapes. La première consiste en la localisation habituelle (transformation de coordonnées + recherche) des services affectés. Ensuite la fonction d'enlèvement de la banque de données est appelée pour chacun d'eux. Finalement, la nouvelle route-horaire est affichée.

5.2.3.8 Module de croisement

Ce module permet d'interchanger deux sections terminales de route-horaire. La localisation des points de croisement s'effectue par une transformation des coordonnées "pixel" reçues en coordonnées (véhicule, heure) et par un parcours dans les listes des routes-horaires visées. Ensuite, il s'agit d'interchanger les numéros de véhicule des services de chaque fin de route. Et finalement, il reste à croiser les pointeurs précédents et suivants de chaque route au point de croisement. Soit SERV1 et SERV2 les pointeurs sur les premiers services de chaque fin de route, le croisement s'effectue comme suit :

```
if (SERV1 → RPREC)
    SERV1 → RPREC → RNEXT = SERV2
else
    ROUTE_HO[SERV1.NOVEH] = SERV2

if (SERV2 → RPREC)
    SERV2 → RPREC → RNEXT = SERV1
else
    ROUTE_HO[SERV2.NOVEH] = SERV1

TEMP = SERV1.RPREC
SERV1.RPREC = SERV2.RPREC
SERV2.RPREC = TEMP.
```

Après ce croisement interne, le programme réaffiche les deux routes-horaires affectées.

5.2.3.9 Module de gestion de l'horaire

Lorsque l'usager utilise l'interface 2 du département Gestion des opérations, ce module s'occupe de la gestion de l'horaire opérationnel affiché dans la moitié supérieure de l'écran. Les fonctions de manipulation (changement de liste, défilement, positionnement, ...) s'effectuent de la même façon qu'au département Planification. La fonction de comparaison entre l'horaire opérationnel et l'horaire cible n'est pas implantée actuellement. Finalement, la fonction EFFACER, qui permet de retourner à l'interface 1 du département, est réalisée en effaçant l'écran et en affichant l'interface 1.

5.2.3.10 Module de graphique ESP-TPS

Ce module comprend toutes les fonctions gérant l'interface de consultation du diagramme espace-temps des routes-horaires. Le tableau V présente les principales fonctions de ce module.

Tableau V : Fonctions de l'interface de consultation

1. Interception
2. Réaffichage
3. Changement
4. Retour

La fonction d'interception capte les entrées (clavier et souris) de l'usager et oriente l'exécution en conséquence. La fonction de réaffichage s'occupe de retracer la route d'un véhicule pointé dans la légende. Il s'agit d'abord d'obtenir le numéro du véhicule à partir des coordonnées du

point indiqué par la souris. La position des éléments de la légende étant fixe et connue, ce numéro de véhicule est rapidement trouvé. Par la suite, la route est retracée grâce à un appel à la fonction graphique appropriée (cf. Appendice B, fct 4).

La fonction de changement permet de remplacer les routes-horaires affichées par celles de d'autres véhicules. Cette fonction n'est pas implantée actuellement.

Finalement, la fonction de retour efface l'écran et affiche l'interface de travail utilisé par le participant.

CONCLUSION

L'approche utilisée pour l'intégration de la dimension "routing-scheduling" au simulateur d'entreprises de transport interurbain de passagers, JET-SET, nous a amené à définir une nouvelle organisation du processus de décision et d'analyse auquel sont confrontés les participants lors d'une séance de simulation. La répartition départementale des décisions permet au participant de saisir clairement les interactions existantes entre les différents domaines d'application des décisions et de prendre ainsi conscience des intérêts différents et souvent conflictuels de chacun. De même, le partage de la responsabilité des différents départements décisionnels entre les participants d'une même équipe permet d'impliquer chaque individu dans une dynamique de communication qui est essentielle à la gestion efficace de l'entreprise simulée.

Sur la base de cette nouvelle organisation du processus de décision et d'analyse, nous avons développé un **système interactif-graphique de prise de décision** comprenant un ensemble d'interfaces de hauts niveaux ainsi que plusieurs outils d'aide à la décision. Étant donné les objectifs, forcément limités, que nous nous étions fixés pour ce stage, nous avons mis l'accent sur les aspects "routing-scheduling", c'est-à-dire sur le développement des interfaces permettant de créer, modifier et consulter une banque de données composée des horaires de service et des horaires publiés de la compagnie, de même que des routes-horaires des véhicules. Notamment, l'interface graphique développée pour la manipulation des routes-horaires propose un ensemble de commandes concis et des outils analytiques-graphiques qui facilitent l'élaboration d'un routing efficace. Finalement, notons que l'utilisation de cette interface pourrait déborder du cadre didactique et analytique du simulateur et être employée à des fins opérationnelles par des planificateurs/gestionnaires.

Le système interactif-graphique de prise de décision inclut aussi un module de génération automatique des horaires publiés consultés par les voyageurs. L'utilisation de la notion de plage horaire pour déterminer les connexions valides et sensées a permis de limiter les horaires publiés aux services indirects logiquement considérés par les voyageurs.

Le simulateur JET-SET/3 n'en est qu'aux premiers stades de son implantation. Il faut d'une part terminer le système interactif-graphique de prise de décision en intégrant les interfaces des autres départements, c'est-à-dire ceux non reliés à la dimension "routing-scheduling", soient marketing, achat/locations et finances. De même, il faut compléter la banque de données en conséquence, c'est-à-dire incorporer les données relatives aux décisions des autres départements. Ces ajouts ayant été prévus dans la conception initiale du système, cette étape devrait s'effectuer aisément. D'autre part, il s'agit évidemment de produire le module de simulation des déplacements.

L'ajout de la dimension opérationnelle au simulateur d'entreprises JET-SET permet aux participants d'expérimenter les trois niveaux de planification/gestion soient les niveaux stratégique, tactique et opérationnel. Il devient ainsi un outil d'autant plus utile qu'il s'adresse à un ensemble plus grand de planificateurs et gestionnaires. Le système final permettra de simuler plus facilement et plus efficacement différentes stratégies d'opération d'une compagnie, entraînant ainsi une amélioration de la qualité de la planification et de la gestion dans le domaine du transport interurbain de passagers.

RÉFÉRENCES

Aho A.V., Hopcroft J.E., Ullman J.D. (1983), *Data Structures and Algorithms*, Addison-Wesley, Massachusetts.

Claybrook B.G. (1983), *File Management Techniques*, John Wiley & Sons, New York.

Elias A.L. (1979), *The Development of an Operational Game for the U.S. Domestic Airline Industry*, FTL Report R78-5, MIT Flight Transportation Laboratory.

Johnson N. (1987), *Advanced Graphics in C: Programming and Techniques*, McGraw-Hill, U.S.A.

Lardinois C., D'Avignon J., Hirou C. (1985), *JET-SET : Jeu Educatif en Transport - Simulation for Education in Transportation*, publication #407, Centre de recherche sur les transports, Université de Montréal.

Lardinois C. (1987), *Simulation and Gaming with JET-SET: An Intercity Passenger Transportation Training Tool*, Simulation & Games 18(1), Sage Publications, 13-33.

Lardinois C. (1988), *Simulation, Gaming and Training in a Competitive Multimodal Multicompany Intercity Passenger Transportation Environment*, publication #588, Centre de recherche sur les transports, Université de Montréal. Accepté pour publication dans le Journal of the Operational Research Society.

Lebeuf D. (1985), *L'Allocation des appareils pour un horaire aérien : une approche interactive-graphique*, publication #415, Centre de recherche sur les transports, Université de Montréal.

Panel Plus (Advanced Screen Manager) (1987), Roundhill Computer Systems Limited, England.

Schildt H. (1987), *C: The Complete Reference*, McGraw-Hill, U.S.A.

Standish T.A. (1980), *Data Structure Techniques*, Addison-Wesley, Massachusetts.

Willumsen L.G., Ortuzar J.D. (1985), *Intuition and Models in Transport Management*, Transportation Research 19A(1), 51-57.

APPENDICE A

Déclaration de la structure de la banque de données

```
     /* enregistrement du service */

struct serv_dir {

   char h_dep[5];              /* heure de depart du service */
   char h_arr[5];              /* heure d'arrivee du service */
   char v_dep;                 /* ville de depart du service */
   char v_arr;                 /* ville d'arrivee du service */
   int no;                     /* numero du service (fichier) */
   int noveh;                  /* numero du vehicule desservant ce service */
   struct serv_dir *rprec;     /* service precedent sur la route du vehicule */
   struct serv_dir *rnext;     /* service suivant sur la route du vehicule */
   struct serv_dir *iprec;     /* activite precedente a la ville de depart */
   struct serv_dir *inext;     /* activite suivante a la ville de depart */
   struct serv_dir *jprec;     /* activite precedente a la ville d'arrivee */
   struct serv_dir *jnext;     /* activite suivante a la ville d'arrivee */
   struct serv_dir *next;      /* service suivant */
   struct serv_dir *prec;      /* service precedent */
   struct serv_ind *gauche;    /* premier service indirect a un transfert
                                  qui contient ce service a gauche */
   struct serv_ind *droite;    /* premier service indirect a un transfert
                                  qui contient ce service a droite */
   struct serv_ind *deux;      /* premier service indirect a deux transfert
                                  contenant ce service */
};



     /* enregistrement du service indirect */

struct serv_ind {

   char h_dep[5];              /* heure de depart du service */
   char h_arr[5];              /* heure d'arrivee du service */
   char v_dep;                 /* ville de depart du service */
   char v_arr;                 /* ville d'arrivee du service */
   int no;                     /* numero du service (fichier) */
   struct serv_dir *serv1;     /* service direct a gauche */
   struct serv_dir *serv2;     /* service direct a droite (ou au centre) */
   struct serv_dir *serv3;     /* service direct a droite (deux trfs) */
   struct serv_ind *prec1;     /* service indirect precedent ayant SERV1 */
   struct serv_ind *next1;     /* service indirect suivant ayant SERV1 */
   struct serv_ind *prec2;     /* service indirect precedent ayant SERV2 */
   struct serv_ind *next2;     /* service indirect suivant ayant SERV2 */
   struct serv_ind *prec3;     /* service indirect precedent ayanr SERV3 */
   struct serv_ind *next3;     /* service indirect suivant ayant SERV3 */
   struct serv_ind *next;      /* service indirect suivant */
   struct serv_ind *prec;      /* service indirect precedent */
};
```

```
     /* structure d'accès aux services et services indirects */

struct index_trf {
   char trf[3];                 /* contient les deux villes de transfert */
   struct serv_ind *first;      /* pointe sur le premier service indirect
                                   ayant ce(s) ville(s) de transferts */
   struct index_trf *next;      /* prochain element dans l'index */
   struct index_trf *prec;      /* element precedent dans l'index */
};


struct serv_dir *mat_sdir [max_ville][max_ville];
                                /* pointeurs sur les premiers services directs
                                   pour chaque paire o/d */

struct serv_dir *route_ho [max_veh];
                                /* pointeurs sur les premiers services directs
                                   de la route de chaque vehicule */

struct serv_dir *activite [max_ville];
                                /* pointeurs sur les premieres activites a
                                   chaque villes */

struct index_trf *mat_sind [max_ville][max_ville];
                                /* pointeurs sur le premier transfert pour
                                   chaque paire o/d */
```

APPENDICE B

Extraits du code

Afin de faciliter la lecture et la compréhension des fonctions (ou extraits) présentées dans cette appendice, les tableaux VI et VII donnent la liste des principales variables et constantes utilisées.

Tableau VI : Variables

Nom	Description
NEXT	Pointeur servant à parcourir des services
SERV	Pointeur sur le service ajouté
BRIS	Tableau indiquant les plages rencontrées
MAX	Heure limite acceptable pour une connexion
DEBORDE	Tableau indiquant les plages valides
NO_FIRST	Entier indiquant le véhicule de la 1ère route affichée

Tableau VII : Constantes

Nom	Description
DEBX_VEH	Coordonnée abscisse du début de l'axe des véhicules
DEBY_VEH	Coordonnée ordonnée du début de l'axe des véhicules
FINX_VEH	Coordonnée abscisse de la fin de l'axe des véhicules
FINY_VEH	Coordonnée ordonnée de la fin de l'axe des véhicules
DEBX_HRE	Coordonnée abscisse du début de l'axe de temps
DEBY_HRE	Coordonnée ordonnée du début de l'axe de temps
FINX_HRE	Coordonnée abscisse de la fin de l'axe de temps
FINY_HRE	Coordonnée ordonnée de la fin de l'axe de temps
DEBX_ESP	Coordonnée abscisse du début de l'axe des villes
DEBY_ESP	Coordonnée ordonnée du début de l'axe des villes
FINX_ESP	Coordonnée abscisse de la fin de l'axe des villes
FINY_ESP	Coordonnée ordonnée de la fin de l'axe des villes
FIRST_HRE	Coordonnée abscisse de la position de la 1ère heure
FIRST_VEH	Coordonnée ordonnée de la position du 1er véhicule
FIRST_ESP	Coordonnée abscisse de la position de la 1ère ville

De plus, la fonction utilitaire CAR permet de transformer le nombre de minutes contenu dans un entier en une chaîne de caractère représentant l'heure correspondante. La fonction MINUTE a l'effet contraire.

fct1 Fonction d'insertion dans la liste de l'horaire de service

```
struct serv_dir *ins_serv(elem,tete)
/* appele par : insere */
/* insere le service elem dans la matrice
   origine/destination */

/* elem : service direct a inserer
   tete : debut de la liste pour cette paire o/d */

struct serv_dir *elem,*tete;

{
   struct serv_dir *cour,*old;  /* sert au parcours et a l'insertion */

   /* si c'est le seul pour cette o/d */
   if (tete == NULL) {
      /* premier service */
      elem->next = NULL;
      elem->prec = NULL;
      return (elem);
   }

   /* sinon on cherche l'endroit en ordre chrono ou l'inserer */
   cour = tete;
   old = NULL;
   while (cour) {
      if (strcmp (cour->h_dep,elem->h_dep) < 0) {
         old = cour;
         cour = cour->next;
      }
      else {
         /* insertion dans le milieu */
         if (cour->prec) {
            cour->prec->next = elem;
            elem->prec = cour->prec;
            elem->next = cour;
            cour->prec = elem;
            return (tete);
         }
         /* insertion au debut de la route */
         elem->next = cour;
         elem->prec = NULL;
         cour->prec = elem;
         return (elem);
      }
   }
   /* insertion a la fin de la route */
   old->next = elem;
   elem->prec = old;
   elem->next = NULL;
   return (tete);
}
```

fct2 Extrait de la fonction pour tester le bris de plages

```
for (i=0; i<=max_vill; i++)
   bris[i] = '0';

if (dir == 'd') {

   /* initialise max avec l'heure de connections maximale */
   car(minute(cour->h_arr) - max_connec,max);

   /* on remonte */
   next = cour->jprec;
   while (next) {

      if (next->v_dep == cour->v_dep)
         /* prochain service avec meme paire o/d -> arret */
         break;
      else
         if (next->v_dep == cour->v_arr) {
            if (strcmp(next->h_dep,max) < 0) break;
            bris[next->v_arr - 'a'] = '1';  /* possibilite de bris */
            next = next->iprec;
         }
         else
            next = next->jprec;
   }

   /* initialise max avec l'heure de connections maximale */
   car(minute(cour->h_arr) + max_connec,max);

   /* on descend */
   next = cour->jnext;
   while (next) {

      if (next->v_dep == cour->v_dep)
         /* prochain service avec meme paire o/d -> arret */
         break;
      else
         if (next->v_dep == cour->v_arr) {

            if (strcmp(next->h_dep,max) > 0) break;

            if ((bris[next->v_arr - 'a'] == '1') &&    /* possibilite */
                (deborde[next->v_arr - 'a'] != '2'))

               /* on elimine les serv. ind. devenus
                  invalides par ce bris */
               elim_sind(next,cour,'d');

            next = next->inext;
         }
         else
            next = next->jnext;
   }
}
```

fct3 Fonction pour transformer une coordonnée

```
int transforme(vehicule,heure,x,y)
/* transforme la coordonnee x,y en numero de
   vehicule et heure selon les axes */
int *vehicule;
char heure[5];
int x,y;
{
   int hauteur = 28;
   int entre = 8;
   int pos_cinq_hre = 300;
   div_t n;
   int temp;


   if (hovisible) y -= 140;
   if (y < 28) return 0;

   n = div(y,hauteur);

   if (n.rem < entre) return 0;

   *vehicule = no_first + n.quot - 1;

   temp = (x - first_hre) * 2;
   temp += pos_cinq_hre;

   car(temp,heure);
   return 1;
}
```

fct4 Fonction pour tracer une route du graphique esp-tps

```
trace_route(veh)
/* trace la route du vehicule veh pour le graphique espace-temps */
int veh;
{
   int temp,x,y,x1,y1;
   int hauteur = 28;
   int pos_cinq_hre = 300;
   char buf[10];
   struct serv_dir *serv;

      serv = route_ho[veh];

      if (serv) {
         /* position de depart */
         x = debx_esp;
         y = first_esp + (hauteur * (serv->v_dep - 'a'));
         /* choisit la couleur en fonction du rang du vehicule affichee
            ex : 1er = noir */
         _setcolor(color_veh[veh - no_first]);

         /* trace la route */
         while (serv) {
            _moveto(x,y);
            x1 = x; y1 = y-1;
            y = first_esp + (hauteur * (serv->v_dep - 'a'));
            temp = minute(serv->h_dep) - pos_cinq_hre;
            x = first_hre + (temp >> 1);

            /* si la route est decrochee on trace la route en pointille */
            if ((serv->rprec) && (serv->v_dep != serv->rprec->v_arr)) {
               _setlinestyle(0xF0F0);
               _lineto(x,y);
               _moveto(x1,y1);
               _lineto(x,y-1);
               _setlinestyle(0xFFFF);
            }
            else {
               _lineto(x,y);
               _moveto(x1,y1);
               _lineto(x,y-1);
            }
            x1 = x; y1 = y-1;

            y = first_esp + (hauteur * (serv->v_arr - 'a'));
            temp = minute(serv->h_arr) - pos_cinq_hre;
            x = first_hre + (temp >> 1);
            _lineto(x,y);
            _moveto(x1,y1);
            _lineto(x,y-1);

            serv = serv->rnext;
         }
         _moveto(x,y);
         _lineto(finx_hre,y);
         _moveto(x,y-1);
         _lineto(finx_hre,y-1);
      }
}
```

fct5 Fonction pour tracer une route du graphique veh-tps

```
aff_route(no)
/* affiche graphiquement la route horaire du vehicule no */
int no;
{
   int courx,coury,temp,tempx,x1,y1,x2,y2,i,j;
   int hauteur = 28;
   int pos_cinq_hre = 300;
   struct serv_dir *cour;
   char buf[10];

   /* on efface la route precedente */
   x1 = debx_veh + 1;
   y1 = first_veh + (hauteur * (no - no_first)) - 20;
   x2 = finx_hre;
   y2 = y1 + 20;

   _setcolor(_getbkcolor());
   _rectangle(_GFILLINTERIOR,x1,y1,x2,y2);
   _setcolor(7);


   /* positionne pour le debut de la route */
   courx = x1 - 1;
   coury = y2;
   tempx = 0;

   cour = route_ho[no];

   /* on ecrit la ville de depart */
   if (cour) {
      _settextposition(coury/14,courx/8 + 2);
      sprintf(buf,"%c",cour->v_dep);
      _outtext(buf);
   }

   /* on dessine chaque service */
   while (cour) {

      _moveto(courx,coury);
      temp = minute(cour->h_dep) - pos_cinq_hre;
      courx = first_hre + (temp>>1);

      /* verifie si route decrochee i.e. conflit de ville */
      if ((cour->rprec) && (cour->v_dep != cour->rprec->v_arr)) {
         _setlinestyle(0xF0F0);
         _lineto(courx,coury);
         _setlinestyle(0xFFFF);
      }
      else
         _lineto(courx,coury);


      _settextposition(coury/14,courx/8 + 2);
      sprintf(buf,"%c",cour->v_dep);
      _outtext(buf);
```

```
        /* verifie si conflit de temps depart avant arrivee */
        if (courx < tempx) {
           _setcolor(12);
           i = (tempx + courx)/2;
           j = coury - 10;
           _moveto(i-6,j-6);
           _lineto(i+6,j+6);
           _moveto(i-6,j+6);
           _lineto(i+6,j-6);
           _setcolor(7);
           _moveto(courx,coury);
        }

        coury -= 20;
        _lineto(courx,coury);

        temp = minute(cour->h_arr) - pos_cinq_hre;
        courx = first_hre + (temp>>1);
        _lineto(courx,coury);

        coury += 20;
        _lineto(courx,coury);
        tempx = courx;

        _settextposition(coury/14,courx/8);
        sprintf(buf,"%c",cour->v_arr);
        _outtext(buf);

        cour = cour->rnext;
     }
     /* termine l'affichage de la route */
     _moveto(courx,coury);
     _lineto(finx_hre,coury);
}
```

REMERCIEMENTS

Je voudrais d'abord remercier mon directeur de maîtrise, Monsieur Christian Lardinois, pour m'avoir offert l'opportunité de réaliser un projet aussi intéressant, de même que Monsieur Teodor G. Crainic pour l'intérêt qu'il a porté à mon stage. Leurs commentaires judicieux ont de plus grandement contribué à améliorer la version finale de ce rapport.

Je tiens également à remercier :

- mon conjoint, Richard Jenkins, pour le support moral et technique qu'il m'a apporté;
- Johanne Côté pour la lecture attentive qu'elle a portée à ce texte, ainsi que pour ses commentaires pertinents;
- Carole Laflamme pour son travail de dactylographie;
- l'organisme F.C.A.R., de même que le Centre de recherche sur les transports pour leur appui financier.